

# Interface Instruction

Model: G1

Version: V1.0

1. Data Collection-----	1
2. Service Access-----	2
2.1 MQTT Access-----	2
2.2 HTTP Access-----	4
2.3 The Detailed Description of Service Access-----	5
3. Steps of Service Access-----	5
3.1 MQTT Communication Mode Instruction-----	6
3.2 Release of Data Collection-----	7
3.3 MQTT Method-----	7
3.4 HTTP Method-----	9
4. Receive Running Command-----	10
4.1 MQTT Method-----	10

## 1. Data Collection

G1 BLE gateway continues collection of BLE broadcast data after launching; if the network is applicable, it sends data to accessed server at frequency of once/second.

The formats of BLE broadcast data that G1 BLE gateway is:

Type	Description
iBeacon	iBeacon protocol as Apple Inc. proposed <a href="https://developer.apple.com/iBeacon/Getting-Started-with-iBeacon.pdf">https://developer.apple.com/iBeacon/Getting-Started-with-iBeacon.pdf</a>
S1 Sensor	S1 temperature and humidity sensor of Minew; please refer to its specification for specific format
Other	Unidentified BLE broadcast data

## 2. Service Access

G1 BLE gateway at present supports MQTT / HTTP network protocol to communicate with local or internet server.

Users may select the network protocol appropriate to their own server framework; MQTT protocol is recommended.

G1 BLE gateway in defaults matches with BeaconYun IoT service and control console for demonstration, you may experience the application of BLE gateway quickly.

### 2.1 MQTT Access

When adopting MQTT access, G1 BLE gateway supports timing uploading of data collection and remote command control, and the message is in JSON format.

G1 BLE gateway supports standard server in MQTT 3.1.1 protocol; please refer to MQTT 3.1.1 protocol file for details, <http://mqtt.org>

Fill the corresponding account information in the service connection setting column of gateway WEB configuration interface, then it takes effect upon saving.



Field	Name	Description
Service Access	Service access method	Select MQTT Client
Upload Interval	Uploading interval	Unit s, 1 in default; 1,2, 3, 4, 5 are optional
Host/IP	MQTT server address	Domain name or IP address of MQTT server
Port	MQTT server port	Port of MQTT server, 1883 in default
Client ID	MQTT Client ID	Client ID defined by MQTT protocol, which is the identification of client
Qos	MQTT QoS grade	QoS grade defined by MQTT protocol QoS=0: one time at most, may repeat or lost QoS=1: one time at least, may repeat or lost QoS=2: only once, guarantee message only reaches once (suitable for strict billing system)
Username	MQTT user name	User name defined by MQTT protocol
Password	MQTT password	Password defined by MQTT protocol
Status Publish Topic	Topic released by data collection	Suitable for timing uploading of data, the message is in JSON format See the section 9.3 for details
Action Control Topic	Topic of control command message	Suitable for receiving control command, the message is in JSON format; currently only support “restarting” command See the section 9.3 for details
Action Control Response Topic	Topic responded by control command	Suitable for responding to control command, the message is in JSON format See the section 9.3 for details





Whether to upload iBeacon	Whether to upload data in type of iBeacon	No in default; YES, NO are optional
Whether to upload S1	Whether to upload data in type of S1	No in default; YES, NO are optional
Whether to upload Gateway	Whether to upload data in type of gateway	No in default; YES, NO are optional
Whether to upload Unknown	Whether to upload data in unknown type	No in default; YES, NO are optional

## 2.2 HTTP Access



When adopting HTTP access, G1 BLE gateway only supports timing uploading of data collection, and the data is in JSON format.


G1 BLE gateway supports Web server in HTTP protocol. Select HTTP Client in the service connection setting column of WEB configuration interface and then fill in the URL information of HTTP server; it takes effect upon saving.


18:14





79

 <http://192.168.99.1/#/service> 

 Status

 Network

 Service

 Other

Service Access

HTTP

Upload Interval

1

Url

http://requestb.in

Whether to upload iBeacon

YES

Whether to upload S1






YES

Whether to upload Gateway


YES

Whether to upload Unknown

NO

18:14

 http://192.168.99.1/#/service

STARTED

NETWORK

SETTINGS

OTHER

Service Access

HTTP

Upload Interval

1

Url

http://requestb.in

Whether to upload iBeacon

YES

Whether to upload S1

YES

Whether to upload Gateway

YES

Whether to upload Unknown


NO


Load Default


Apply

<

>







Field	Name	Description
Service Access	Service access method	Select HTTP Client
Upload Interval	Upload Interval	Unit s, 1 in default; 1,2, 3, 4, 5 are optional
URL	URL for data collection	It is used for uploading data regularly with the beginning of http://, data is the format JSON by POST way. More details, see the below section 7.3.
Whether to upload iBeacon	Whether to upload data in type of iBeacon	YES in default; YES, NO are optional
Whether to upload S1	Whether to upload data in type of S1	YES in default; YES, NO are optional
Whether to upload Gateway	Whether to upload data in type of gateway	YES in default; YES, NO are optional
Whether to upload Unknown	Whether to upload data in unknown type	No in default; YES, NO are optional

## 2.3 The Detailed Description of Service Access

The service access of G1 BLE gateway needs the service support from background MQTT or HTTP. Take BeaconYun IoT as an example to explain the service access in details.

## 3. Steps of Service Access

The following steps are for the gateway access in the BeaconYun IoT service

Create a gateway in the IoT services console and register your gateway at the same time. The gateway creates a gateway MAC address (the only identity gateway) and a gateway name (the default is "GW-" + MAC gateway, where they are hexadecimal capital character, in short, as gateway Name, which can be seen on the console)

User inputs the relative gateway parameters into the gateway, the device is valid immediately afterwards.

The modes for G1 BLE gateway accessing BeaconYun IoT are MQTT and HTTP, the default one is MQTT, parameters can be set as the following table.

MQTT access parameters to BeaconYun IoT service (the gateway is in default)

Field	Names	BeaconYun IoT Service parameters
Service Access	Service access method	MQTT Client
Upload Interval	Upload Interval	Unit s, 1 in default; 1,2, 3, 4, 5 are optional
Host/IP	MQTT service address	iot.beaconyun.com
Port	MQTT server port	1883
Client ID	MQTT Client ID	`\${gatewayMAC}` is the gateway Mac, where they are hexadecimal lowercase characters, for example: aabbccddeeff
Qos	MQTT QoS grade	0 (or 1)
Username	MQTT user name	(blank)

Password	MQTT password	(blank)
Status Publish Topic	Topic released by data collection	$\{gatewayMac\}$ is the gateway Mac, where they are hexadecimal lowercase characters, for example: /gw/aabbccddeeff/status
Action Control Topic	Topic of control command message	$\{gatewayMac\}$ is the gateway Mac, where they are hexadecimal lowercase characters, for example: /gw/aabbccddeeff/action
Action Control Response Topic	Topic responded by control command	$\{gatewayMac\}$ is the gateway Mac, where they are hexadecimal lowercase characters, for example: /gw/aabbccddeeff/action/response.
Whether to upload iBeacon	Whether to upload data in type of iBeacon	YES in default; YES, NO are optional
Whether to upload S1	Whether to upload data in type of S1	YES in default; YES, NO are optional
Whether to upload Gateway	Whether to upload data in type of gateway	YES in default; YES, NO are optional
Whether to upload Unknown	Whether to upload data in unknown type	No in default; YES, NO are optional

Note: the current support message QOS is 0,1, so far, the 2 is not support due to big cost in device.

Access parameters of HTTP to BeaconYun IoT service

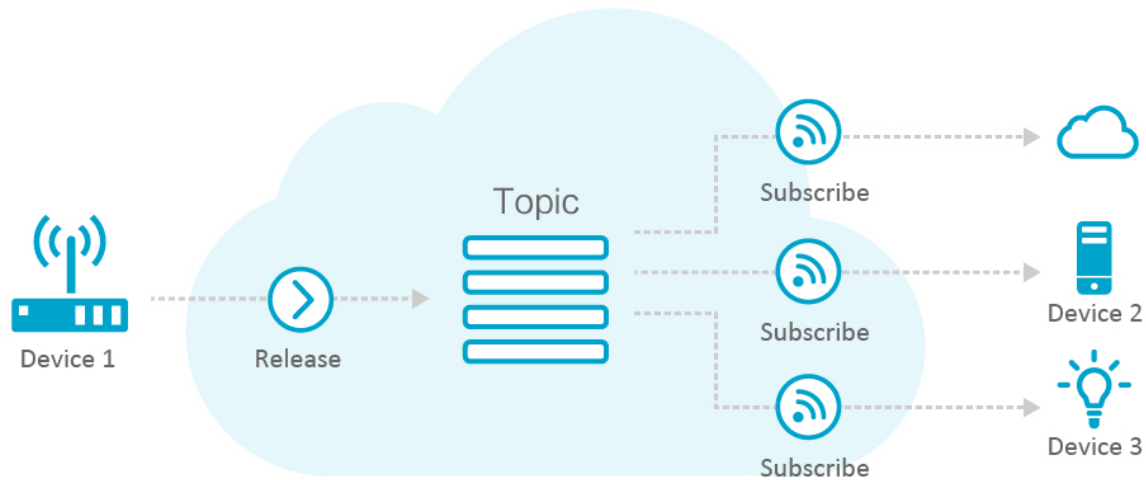
Field	Name	BeaconYun IoT Service parameter
Service Access	Service access method	HTTP Client
Upload Interval	Uploading interval	Unit s, 1 in default; 1, 2, 3, 4, 5 are optional
Status Post URL	URL released by data collection	$\{gatewayMac\}$ is the gateway Mac, where they are hexadecimal lowercase characters, for example: <a href="http://iot.beaconyun.com/gw/aabbccddeeff/status">http://iot.beaconyun.com/gw/aabbccddeeff/status</a> .
Whether to upload iBeacon	Whether to upload data in type of iBeacon	No in default; YES, NO are optional
Whether to upload S1	Whether to upload data in type of S1	No in default; YES, NO are optional
Whether to upload Gateway	Whether to upload data in type of gateway	No in default; YES, NO are optional
Whether to upload Unknown	Whether to upload data in unknown type	No in default; YES, NO are optional

With respect to the operation instruction of BeaconYun IoT, please see file Thingoo G1 Dashboard User Guide.

### 3.1 MQTT Communication Mode Instruction

Communication mode provided by MQTT is based on the release/subscribe mode of Topic.

Release/subscribe is based on the message forward of Topic by route, the release and subscribe can be carried out both at device end and service end, so that the asynchronous communication can be achieved eventually. (see the following picture)



The Topic format must be layered by "/" to distinguish each category. Category names can only contain letters, numbers and underscores (\_) in each named category, each category cannot be an empty one.

In BeaconYun IoT service, the first two categories are set already; the first one is fixed string gw, the second category stands for the gateway MAC address. Topic belongs to its corresponding gateway, which cannot be used by other gateways for release/subscribe communication.

## 3.2 Release of Data Collection

BLE data collected by G1 BLE gateway and its status data are released to the server by MQTT or HTTP, the data format is in type of JSON array, release frequency is once per 1 second.

## 3.3 MQTT Method

Data is sent to designated Topic in MQTT service by MQTT method.

Different MQTT service provider provides different Topic in accordance with its own file. To self-developed MQTT service, the user can see our design for reference. Take BeaconYun IoT as an example, the definition of release for data collection is:

Topic	Operation	Description
/gw/\${gatewayMac}/status	Release	Topic receives BLE data uploaded by gateway

In which, \${gatewayMac} is the gateway Mac, where they are hexadecimal lowercase characters.

### Message example

```
[
  {
    "timestamp": "2017-04-28T08:16:13+08:00",
    "type": "iBeacon",
    "mac": "CC0101000011",
    "bleName": "MiniBeacon_00012",
    "ibeaconUuid": "FDA50693A4E24FB1AFCFC6EB07647825",
    "ibeaconMajor": 10001,
    "ibeaconMinor": 19641,
    "ibeaconTxPower": -58,
    "rssi": -76,
```

```

    "battery": 100,
  },
  {
    "timestamp": "2017-04-28T08:16:14+08:00",
    "type": "S1",
    "mac": "CC3101000034",
    "bleName": "S1",
    "rssi": -76,
    "battery": 100,
    "temperature": 21.23,
    "humidity": 43.23
  },
  {
    "timestamp": " 2017-04-28T08:16:14+08:00",
    "type": "Unknown",
    "mac": "EF3101000034",
    "bleName": "MI",
    "rssi": -76,
    "rawData": "3A4E24FB1AFCFC6EB07647825FDA50693A4E24FB1AFCFC6EB07647825"
  },
  {
    "timestamp": " 2017-04-28T08:16:14+08:00",
    "type": "Gateway",
    "mac": "EF3101000034",
    "bleName": "GW-EF3101000034",
  }
]

```

The detailed description of message data fields is as follows:

Field	Names	Description
Timestamp	Timestamp	Time format (ISO 8601)
Type	Type	The type can be iBeacon、S1、Unknown (unknown)、Gateway
MAC	Device's MAC	MAX is hexadecimal capital characters
BE Name	Device's BLE name	It is"" if it does not exist.
iBeacon UUID	Device's UUID	It exists if the device is iBeacon
iBeacon Major	Device's Major	It exists if the device is iBeacon
iBeacon Minor	Device's Minor	It exists if the device is iBeacon
RSSI	Device's RSSI	It exists if the device is iBeacon
iBeacon Tx Power	Device's Tx Power	It exists if the device is iBeacon
Battery	Device's power capacity	It exists if the device is iBeacon or S1 sensor
Temperature	Device's temperature	It exists if the device is S1 sensor
Humidity	Device's humidity	It exists if the device is S1 sensor
RawData	BLE broadcast data	BLE broadcast data is hexadecimal capital characters It exists if the device is Unknown



### 3.4 HTTP Method

Data is POSTED to a designated URL by HTTP method.

To self-developed HTTP server, the user can see our design for reference, take Beacon Yun IoT as an example, the definition of release for data collection is:

URL	Operation	Description
http://iot.beaconyun.com/gw/\${gatewayMac}/status	Release	Topic receives BLE data uploaded by gateway

In which, \${gatewayMac} is the gateway Mac, where they are hexadecimal lower case characters.

#### Message example

```
[
  {
    "timestamp": "2017-04-28T08:16:13+08:00",
    "type": "iBeacon",
    "mac": "CC0101000011",
    "bleName": "MiniBeacon_00012",
    "ibeaconUuid": "FDA50693A4E24FB1AFCFC6EB07647825",
    "ibeaconMajor": 10001,
    "ibeaconMinor": 19641,
    "ibeaconTxPower": -58,
    "rssi": -76,
    "battery": 100,
  },
  {
    "timestamp": "2017-04-28T08:16:14+08:00",
    "type": "S1",
    "mac": "CC3101000034",
    "bleName": "S1",
    "rssi": -76,
    "battery": 100,
    "temperature": 21.23,
    "humidity": 43.23
  },
  {
    "timestamp": " 2017-04-28T08:16:14+08:00",
    "type": "Unknown",
    "mac": "EF3101000034",
    "bleName": "MI",
    "rssi": -76,
    "rawData": "3A4E24FB1AFCFC6EB07647825FDA50693A4E24FB1AFCFC6EB07647825"
  },
  {
    "timestamp": " 2017-04-28T08:16:14+08:00",
    "type": "Gateway",
    "mac": "EF3101000034",
    "bleName": "GW-EF3101000034",
  }
]
```

The detailed description of message data fields is as follows:

Field	Names	Description
Timestamp	Timestamp	Time format (ISO8601)
Type	Type	Can be iBeacon、S1、Unknown (unknown), Gateway
MAC	Device's MAC	MAX is hexadecimal capital characters
BLEName	Device's BLE name	It is "" if it does not exist.
iBeaconUUID	Device's UUID	It exists if the device is iBeacon
iBeaconMajor	Device's Major	It exists if the device is iBeacon
iBeaconMinor	Device's Minor	It exists if the device is iBeacon
RSSI	Device's RSSI	It exists if the device is iBeacon
iBeacon Tx Power	Device's Tx Power	It exists if the device is iBeacon
Battery	Device's power capacity	It exists if the device is iBeacon or S1 sensor
Temperature	Device's temperature	It exists if the device is S1 sensor
Humidity	Device's humidity	It exists if the device is S1 sensor
RawData	BLE broadcast data	BLE broadcast data is hexadecimal capital characters It exists if the device is Unknown

## 4. Receive Running Command

The G1 BLE gateway can subscribe the Topic to receive control command by MQTT, after receiving the command, the gateway responds to commands through another Topic release command and performs these commands; the data format is also JSON array. This function is not supported in the HTTP method.

### 4.1 MQTT method

In MQTT mode, the gateway subscribes to the Topic designated by the MQTT service, after receiving the control command, the gateway responds to this message by sending a control command by another Topic.

Different MQTT service provider provides different Topic in accordance with its own file. To self-developed MQTT service, the user can see our design for reference. Take BeaconYun IoT as an example, the definition of release for data collection is:

Topic	Operation	Description
/gw/\${gatewayMac}/action	Subscribe	Gateway receives the control commands sent by IoT service
/gw/\${gatewayMac}/action/response	Release	The feedback information is sent by gateway after receiving the control command

Where, \${gatewayMac} is the gateway Mac, where they are hexadecimal lowercase characters.

### Message example

The example of request information of control command is as follows:

```
{ "action": "reboot", "request Id": "b2c3d4e5-3424-4dca-32dc-12b73290cfed" }
```

The detailed description of message data fields is as follows:

Field	Name	Description
Action	Operation	Control command, currently support "reboot"
Request ID	Request ID	UUID format, lowercase characters

Example of responding message of control command.

```
{ "code": 200, "message": "success", "request Id": "b2c3d4e5-3424-4dca-32dc-12b73290cfed" }
```

Or

```
{ "code": 400, "message": "unknown error", "request Id": "b2c3d4e5-3424-4dca-32dc-12b73290cfed" }
```

The detailed description of message data fields is as follows:

Field	Name	Description
Code	Response code	200 (stands for success) or 400 (stands for fault)
Message	Response information	Success (stands for success) or unknown error (stands for unknown error)
Request ID	Request ID	UUID format, it is used to match request and response information

Note: It is subject to any changes without prior notification. MINEW reserves the final right of interpretation.