

服务接口说明

**目录**

目录

[1. 前言 5](" \l "_Toc120095460)

[2. 接入说明及数据格式 5](" \l "_Toc120095461)

[2.1 范围 5](" \l "_Toc120095462)

[2.2 规范性引用文件 5](" \l "_Toc120095463)

[2.3 术语和定义、缩略语 5](" \l "_Toc120095464)

[2.3.1 术语和定义 5](" \l "_Toc120095465)

[2.4 文档通用信息 6](" \l "_Toc120095466)

[2.4.1 返回结果格式 6](" \l "_Toc120095467)

[2.5 LBS接口说明 7](" \l "_Toc120095468)

[2.5.1 LBS定位接口 7](" \l "_Toc120095469)

[2.6 设备服务注册接口说明 11](" \l "_Toc120095470)

[2.6.1 设备服务校验设备状态与profile接口 11](" \l "_Toc120095471)

[2.7 签名生成 13](" \l "_Toc120095472)

[2.7.1 构造签名摘要 13](" \l "_Toc120095473)

[2.7.2 生成签名串 14](" \l "_Toc120095474)

[2.7.3 设置HTTP头 15](" \l "_Toc120095475)

[2.8 附录 16](" \l "_Toc120095476)

[2.8.1 报文主体组装排列顺序 16](" \l "_Toc120095477)

[2.8.2 签名算法 17](" \l "_Toc120095478)

[2.8.3 随机数算法 20](" \l "_Toc120095479)

变更记录

|  |  |  |  |
| --- | --- | --- | --- |
| **变更人** | **日期** | **变更说明** | **协议版本** |
| 蔡志雄 | 2021/02/03 | LBS接口文档初始版本 | 1.0 |
| 蔡志雄 | 2022/03/24 | 添加文档必要说明、排版调整 | 2.0 |
| 蔡志雄 | 2022/10/09 | location/lbs入参新增字段,可以选择定位源以及优先级 | 2.0 |
| 陈长峰 | 2022/11/03 | 添加/devices/verifyServiceStatusAndIccid认证接口文档 | 2.0 |

# 前言

本指引文件旨在帮助设备接入广通亿联公司LBS服务。设备通过访问LBS服务接口，获取经纬度信息。

# 接入说明及数据格式

## 范围

本规范规定了广通亿联公司LBS服务与设备之间的通讯协议与数据格式。

## 规范性引用文件

HTTP/HTTPS协议规范。

## 术语和定义、缩略语

### **2.3.1 术语和定义**

1. HTTP

HTTP协议是Hyper Text Transfer Protocol（超文本传输协议）的缩写,是用于从万维网（WWW:World Wide Web ）服务器传输超文本到本地浏览器的传送协议。HTTP是一个基于TCP/IP通信协议来传递数据。

## 文档通用信息

### 返回结果格式

1. HTTP响应状态码（常见的）

|  |  |  |  |
| --- | --- | --- | --- |
| **序号** | **Code** | **Message** | **备注** |
| 1 | 200 | ok | http请求成功 |
| 2 | 401 | Unauthorized | 1. null cert chain  2. unknown certificate |
| 3 | 404 | Not Found | url资源未发现 |
| 4 | 500 | Internal Server Error | 内部服务错误 |
| 5 | 502 | Bad Gateway | 网关错误 |

1. 通用返回参数

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **序号** | **字段名称** | **字段含义** | **可空** | **长度** | **备注** |
| 1 | code | 返回码 |  |  |  |
| 2 | message | 返回信息 |  |  |  |
| 3 | data | 返回内容 |  |  |  |

1. 返回码

|  |  |  |  |
| --- | --- | --- | --- |
| **序号** | **Code** | **Message** | **备注** |
| 0 | -1 | Gateway Error; unknown error | 内部网关错误；未知错误（理论上不应该出现未知错误） |
| 1 | 0 | success | 成功 |
| 2 | 9000 | request parameter validation failed | 请求参数校验失败 |
| 3 | 9001 | request parameters is null or json parse error | 请求参数为空或者json无法解析 |
| 4 | 9002 | Internal service path error | 内部服务错误 |
| 5 | 9003 | request wifi access points less than 2 | Wifi节点数不足2个 |
| 6 | 9004 | wifi mac address Invalid | Wifi节点的mac地址错误 |
| 7 | 9006 | cell tower and wifi param all null | 蜂窝基站与wifi节点都没有传数据 |
| 8 | 9007 | Unable to parse location information with your parameters | 传入的参数无法解析到位置数据 |
| 9 | 10001 | Validation failed: [%s] | 请求参数校验错误，例：[nonce\_str|signature|timestamp] required or format error |
| 10 | 10007 | [%s] non-existent | 例：sn不存在；device secret不存咋 |
| 11 | 10009 | [%s] cannot be empty, please check and try again | 例：请求头Authorization不能为空 |
| 12 | 14109 | request invalidation | 请求过期，默认为五分钟有效期 |
| 13 | 14110 | signature verification failed | 验签失败 |
| 14 | 14111 | wrong authentication type | 请求头Authorization认证类型错误 |

## LBS接口说明

### LBS定位接口

|  |  |
| --- | --- |
| 接口URL | https://{{company}}.hgw.thingsmatrix.io/device/v{apiVersion}/location/lbs/{sn} |
| 调用方式 | POST |
| 请求头 | **[详见签名生成](#_签名生成)** |
| 接口描述 | 使用蜂窝基站、WiFi节点信息获取定位数据接口 |
| 接口类型 | 查询 |

1. **输入参数:采用json格式**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 序号 | 字段名 | 数据类型 | 长度 | 可空 | 字段说明 | 备注 |
| 1 | cellTowers | 对象数组 | 无 | Y | 蜂窝基站数据 |  |
| 2 | wifiAccessPoints | 对象数组 | 无 | Y | Wifi节点数据 | 只有wifiAccessPoints数据时，必须大于等于两个wifi节点 |
| 3 | radioType | String | 无 | Y | 移动无线电类型 | "gsm", "cdma", "wcdma", "lte", "nr"(其他值默认转成gsm) |
| 4 | sn | String | 无 | Y | 客户端唯一标识序列号 |  |
| 5 | lbsPriority | Number | 无 | Y | 定位源优先级 | 0-amap/google，1-google/amap，2-google，3-amap |

cellTowers下的对象参数：

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 序号 | 字段名 | 数据类型 | 长度 | 可空 | 字段说明 | 备注 |
| 1 | cid | Number | 无 | N | cell的唯一标识符 |  |
| 2 | mcc | Number | 无 | N | 移动国家代码 |  |
| 3 | mnc | Number | 无 | N | 移动网络代码 |  |
| 4 | lac | Number | 无 | N | GSM和WCDMA网络的LAC (Location Area Code)。CDMA网络的NID (Network ID) |  |
| 5 | signalStrength | Number | 无 | Y | 无线电信号强度测量dBm |  |

wifiAccessPoints：

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 序号 | 字段名 | 数据类型 | 长度 | 可空 | 字段说明 | 备注 |
| 1 | macAddress | String | 无 | Y | WiFi节点的MAC地址 | 通常被称为BSS、BSSID或MAC地址。分隔符必须为:(冒号 |
| 2 | signalStrength |  | 无 | Y | 用dBm测量的电流信号强度 |  |

请求参数补充说明：

|  |
| --- |
| "cellTowers"、"wifiAccessPoints"为可选字段，为提高定位精度，尽可能提供  对于"cellTowers"，如果传入"cellTowers"，其属性"cid"、"mcc"、"mnc"必传  对于"wifiAccessPoints"，如果传入"wifiAccessPoints"， "macAddress"必传，其他可选。如果cellTowers不传，wifiAccessPoints必须大于等于两个wifi节点。 |

1. **输出参数:采用json格式**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 字段名 | 数据类型 | 字段说明 | 备注 |
| 1 | code | INT | 结果码 |  |
| 2 | message | STRING | 结果说明 |  |
| 3 | data | OBJECT | 数据体 |  |
| 4 | data.location | OBJECT | 经纬度数据 |  |
| 5 | data.accuracy | INT | 定位精度半径 |  |
| 6 | data.location.lat | DOUBLE | 纬度 |  |
| 7 | data.location.lng | DOUBLE | 经度 |  |

1. **请求示例**

接口url：https://hpdemo.hgw.thingsmatrix.io/device/v2/location/lbs/2022MQTTS

POST输入示例：

|  |
| --- |
| {  "cellTowers": [  {  "cid": 170402199,  "lac": 35632,  "mcc": 310,  "mnc": 410,  "signalStrength": -60  }  ],  "wifiAccessPoints": [  {  "macAddress": "84:d4:7e:09:a5:f1",  "signalStrength": -43  },  {  "macAddress": "44:48:c1:a6:f3:d0",  "signalStrength": -55  }  ]  } |

1. **返回结果示例：**

|  |
| --- |
| {  "code": 0,  "message": "ok",  "data": {  "location": {  "lat": 22.575175,  "lng": 113.943605  },  "accuracy": 1198  }  } |

## 设备服务注册接口说明

### 设备服务校验设备状态与profile接口

|  |  |
| --- | --- |
| 接口URL | https://{{company}}.hgw.thingsmatrix.io/device/v{apiVersion}/inventory/devices/{sn}/verifyServiceStatusAndIccid |
| 调用方式 | POST |
| 请求头 | [详见签名生成](#_签名生成) |
| 接口描述 | 根据设备上报sn和iccid，返回设备端校验结果 |
| 接口类型 | 查询 |

1. **输入参数:采用json格式**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 序号 | 字段名 | 数据类型 | 长度 | 可空 | 字段说明 | 备注 |
| 1 | sn | String | 无 | N | 客户端唯一标识序列号 |  |
| 2 | iccid | String | 无 | Y | Esim虚拟卡号 |  |

请求参数补充说明：

|  |
| --- |
| https://{{company}}.hgw.thingsmatrix.io/device/v{apiVersion}/inventory/devices/{sn}/verifyServiceStatusAndIccid  {{company}}：公司自定义域名  apiVersion：api版本。目前为2  sn：设备序列号 |

1. **输出参数:采用json格式**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 字段名 | 数据类型 | 字段说明 | 备注 |
| 1 | code | INT | 结果码 |  |
| 2 | message | STRING | 结果说明 |  |
| 3 | data | OBJECT | 数据体 |  |
| 4 | data.status | INT | 设备状态 0:DEACTIVATED,1: ACTIVATED | 必须 |
| 5 | data.operation | INT | Profile操作码  0：不用操作  1: 安装新profile(删除旧profile)  2：删除 | 必须 |
| 6 | data.token | STRING | Activation code(去下载profile)  operation的值为1是，才有返回值 | 非必须 |
| 7 | data.simMode | INT | 设备sim模式：0.iot ; 1.user sim only | 必须 |

1. **请求示例**

接口url：http://hpdemo.hgw.thingsmatrix.io/device/v2/inventory/devices/2022MQTTS/verifyServiceStatusAndIccid

POST输入示例：

|  |
| --- |
| {  "sn": "2022MQTTS",  "iccid": "8935711001000505070"  } |

1. **返回结果示例：**

|  |
| --- |
| {  "code": 0,  "message": "ok",  "data": {  "status": 0,  "operation": 2  }  } |

## 签名生成

### 构造签名摘要

我们希望客户端的技术开发人员按照当前文档约定的规则构造签名摘要。ThingsMatrix会使用同样的方式构签名摘要，如果客户端构造签名摘要的方式错误，将导致签名验证不通过，下面先说明签名摘要的具体格式。

签名摘要一共分为五个部分，每一部分为一个参数。参数拼接以&（ASCII编码值为0x26）分隔。

HTTP请求方法&URL&请求时间戳&请求随机串&请求报文主体

我们通过”设备服务校验设备状态与profile”接口，一步一步想开发者介绍如何进行请求签名。按照接口文档，获取”设备服务校验设备状态与profile”的URL例：<http://hpdemo.hgw.thingsmatrix.io/device/v2/inventory/devices/2022MQTTS/verifyServiceStatusAndIccid>请求方法为POST。

第一步，获取HTTP请求的方法(GET，POST，PUT)等。例：

POST

第二步，获取请求的绝对URL，并去除域名和请求context path部分得到参与签名的URL。如果请求中有查询参数，URL末尾应附加有’?’和对应的查询字符串（**querystring查询参数的顺序按照参数名ASCII码从小到大排序**）。例：

/v2/inventory/devices/2022MQTTS/verifyServiceStatusAndIccid

第三步，获取发起请求时的系统当前时间戳，即格林威治1970年01月01日00时00分00秒起至现在的总秒数，作为请求时间戳。ThingsMatrix会拒绝处理很久之前发起的请求，请客户端保持自身系统的时间准确。例：

1668077215

第四步，生成一个请求随机串，可参见生成**[附录随机数算法](" \l "_随机数算法)**。

ibuaiVcKdpRxkhJA

第五步，获取请求中的请求报文主体（request body），报文主体组装排列顺序详见**[附录报文主体组装排列顺序](" \l "_报文主体组装排列顺序)**。

* 请求方法为GET时，报文主体为空
* 当请求方法为POST或PUT时，请使用真实发送的JSON报文

iccid=8935711001000505070&sn=2022MQTTS

第六步，按照前述规则，构造的请求签名摘要为：

POST&/v2/inventory/devices/2022MQTTS/verifyServiceStatusAndIccid&1668077215&ibuaiVcKdpRxkhJA&iccid=8935711001000505070&sn=2022MQTTS

### 生成签名串

绝大多数编程语言提供的签名函数支持对签名摘要进行签名。强烈建议客户端技术开发人员调用该类函数，使用设备配置的私钥对待签名摘要进行SHA256（目前ThingsMatrix支持*HMAC-SHA256*）签名，并对签名结果进行Base64编码得到签名值，详见**[附录签名算法](" \l "_附录)**。

第一步：使用上一步骤构造的签名摘要，配置设备密钥通过签名算法生成sign串，即hash\_hmac(‘sha256’, ‘签名摘要’ + ‘&key=设备密钥’, key)，例

*Hash\_hmac*(‘sha256’, ‘POST&/v2/inventory/devices/2022MQTTS/verifyServiceStatusAndIccid&1668077215&ibuaiVcKdpRxkhJA&iccid=8935711001000505070&sn=2022MQTTS&key=b06c75b58d1701ff470119a4114f8b45’,’b06c75b58d1701ff470119a4114f8b45’) = ‘3de326adb7d3ea1ebbafb37bc4ca3e16d73e441cb3243415d727c73aba0ed424’

第二步：再将sign串转换为大写串，即sign.toUpperCase()，例：

3DE326ADB7D3EA1EBBAFB37BC4CA3E16D73E441CB3243415D727C73ABA0ED424

第三步：最后，对第二步生成的串进行Base64转换得到我们所需的签名串，例：

M0RFMzI2QURCN0QzRUExRUJCQUZCMzdCQzRDQTNFMTZENzNFNDQxQ0IzMjQzNDE1RDcyN0M3M0FCQTBFRDQyNA==

### 设置HTTP头

ThingMatrix API要求请求通过HTTP Authorization头来传递签名。Authorization由认证类型和签名信息两个部分组成。格式如下：

Authorization: 认证类型 签名信息

具体组成为：

* 认证类型，目前为THINGSMATRIX-HMAC-SHA256
* 签名信息。
* 请求随机串nonce\_str
* 时间戳timestamp
* 签名值signature

备注：以上三项签名信息，无顺序要求

Authorization 头的示例如下：（注意，示例因为排版可能存在换行，实际数据应在一行）

Authorization: THINGSMATRIX-HMAC-SHA256 nonce\_str=ibuaiVcKdpRxkhJA,timestamp=1668077215, signature= M0RFMzI2QURCN0QzRUExRUJCQUZCMzdCQzRDQTNFMTZENzNFNDQxQ0IzMjQzNDE1RDcyN0M3M0FCQTBFRDQyNA==

最终我们可以组装一个包含了签名的HTTP请求，例：

curl --location --request POST 'http://hpdemo.hgw.thingsmatrix.io/device/v2/inventory/devices/2022MQTTS/verifyServiceStatusAndIccid' \

--header ' Authorization: THINGSMATRIX-HMAC-SHA256 nonce\_str=ibuaiVcKdpRxkhJA,timestamp=1668077215,signature=M0RFMzI2QURCN0QzRUExRUJCQUZCMzdCQzRDQTNFMTZENzNFNDQxQ0IzMjQzNDE1RDcyN0M3M0FCQTBFRDQyNA==' \

--header 'Content-Type: application/json' \

--data-raw '{

"sn":"2022MQTTS",

"iccid": " 8935711001000505070"

}'

## 附录

### 报文主体组装排序顺序

设所有发送或者接收到的数据为集合M，将集合M内非空参数值的参数按照参数名ASCII码从小到大排序（字典序），使用URL键值对的格式（即key1=value1&key2=value2…）。

**特别注意以下重要规则：**

◆ 参数名ASCII码从小到大排序（字典序），如果集合M出现相同的参数名（例：JSONArray中），则按照其值的ASCII码从小到大再排序；

◆ 如果参数的值为空不参与签名（空参数值指的是空字符串，如果为null，仍然需要纳入签名摘要，例：&iccid=null）；

◆ 参数名区分大小写；

◆ 接口可能增加字段，验证签名时必须支持增加的扩展字段;

◆ 如果Request Body较复杂，需要将JSON扁平化处理，再生成签名串，例：

{"a":1,"b":{"b1":2},"c":[1,2,3,4],"d":{"d1":{"d1\_1":100,"d1\_2":101},"d2":["d2\_1","d2\_2"]},"e":[1,2]}

扁平化处理后：

a:1

b.b1:2

c:1

c:2

c:3

c:4

d.d1.d1\_1:100

d.d1.d1\_2:101

d.d2:d2\_1

d.d2:d2\_2

e:1

e:2

### 签名算法

签名生成步骤如下：

stringA=HTTP请求方法&URL&请求时间戳&请求随机串&请求报文主体，在stringA最后拼接上设备密钥key得到stringSignTemp字符串，并对stringSignTemp进行HMAC-SHA256运算，再将得到的字符串所有字符转换为大写，得到sign值signValue，最后对sign进行Base64编码得到最终签名值。

举例：

**示例1请求如下：**

curl --location --request POST 'http://hpdemo.hgw.thingsmatrix.io/device/v2/inventory/devices/2022MQTTS/verifyServiceStatusAndIccid' \

--header 'Content-Type: application/json' \

--data-raw '{

"sn":"2022MQTTS",

"iccid": "8935711001000505070"

}'

第一步：对参数按照key=value的格式，并按照参数名ASCII字典序排序如下：

stringA = "POST&/v2/inventory/devices/2022MQTTS/verifyServiceStatusAndIccid&1668077215&ibuaiVcKdpRxkhJA&iccid=8935711001000505070&sn=2022MQTTS";

第二步：拼接API密钥：

HMAC-SHA256签名方式：

stringSignTemp=stringA+"&key=b06c75b58d1701ff470119a4114f8b45" //注：key设备库存为设备配置的私钥key

sign=hash\_hmac("sha256",stringSignTemp,key).toUpperCase()="3DE326ADB7D3EA1EBBAFB37BC4CA3E16D73E441CB3243415D727C73ABA0ED424" //注：HMAC-SHA256签名方式，部分语言的hmac方法生成结果二进制结果，需要调对应函数转化为十六进制字符串。

第三步：对生成的sign进行Base64编码

Base64(sign)= M0RFMzI2QURCN0QzRUExRUJCQUZCMzdCQzRDQTNFMTZENzNFNDQxQ0IzMjQzNDE1RDcyN0M3M0FCQTBFRDQyNA==

**示例2请求如下：**

curl --location --request POST 'https://hpdemo.hgw.thingsmatrix.co/device/v2/location/lbs/2022MQTTS' \

--header 'Content-Type: application/json' \

--data-raw '{

"cellTowers": [

{

"cid": 170402199,

"lac": 35632,

"mcc": 310,

"mnc": 410,

"signalStrength": -60

}

],

"wifiAccessPoints": [

{

"macAddress": "84:d4:7e:09:a5:f1",

"signalStrength": -43

},

{

"macAddress": "44:48:c1:a6:f3:d0",

"signalStrength": -55

}

]

}'

第一步：对参数按照key=value的格式，并按照参数名ASCII字典序排序如下：

stringA = "POST&/v2/location/lbs/2022MQTTS&1668077215&ibuaiVcKdpRxkhJA&cellTowers.cid=170402199&cellTowers.lac=35632&cellTowers.mcc=310&cellTowers.mnc=410&cellTowers.signalStrength=-60&wifiAccessPoints.macAddress=44:48:c1:a6:f3:d0&wifiAccessPoints.macAddress=84:d4:7e:09:a5:f1&wifiAccessPoints.signalStrength=-43&wifiAccessPoints.signalStrength=-55";

第二步：拼接API密钥：

HMAC-SHA256签名方式：

stringSignTemp=stringA+"&key=b06c75b58d1701ff470119a4114f8b45" //注：key设备库存为设备配置的私钥key

sign=hash\_hmac("sha256",stringSignTemp,key).toUpperCase()="245C9359BA0E47D07B3D47A365157F2C2F83A87BCC47A33B2FE38AE63E8E9377" //注：HMAC-SHA256签名方式，部分语言的hmac方法生成结果二进制结果，需要调对应函数转化为十六进制字符串。

第三步：对生成的sign进行Base64编码

Base64(sign)= MjQ1QzkzNTlCQTBFNDdEMDdCM0Q0N0EzNjUxNTdGMkMyRjgzQTg3QkNDNDdBMzNCMkZFMzhBRTYzRThFOTM3Nw==

### 随机数算法

ThingsMatrix API接口协议中包含的字段nonce\_str，主要保证签名不可预测。我们推荐生成随机数算法如下：调用随机数函数生成，将得到的值转换为字符串。