Totarget GPS Platform Integration Guide

Overview

As a third-party partner, you're super easy to receive the live data from us and send the command to the particular hardware device.

Here is the convention and terminology in the following guide:

- **TGP** - **T**otarget **G**PS **P**latform which means us
- **TIS** - **T**hird-party **I**ntegration **S**ystem which means you or the exists backend
- **HTTP** - The **HTTP** we mentioned in the entire guide, it includes the **HTTP** and **HTTPS**

We'll send you an *HTTP post request* to you for pushing the real-time hardware device response message, and you can send us an *HTTP post request* for sending the ELock command to the particular hardware device.

How does it works

1. First, as the **TIS**, you need to contact us to register the below information:
    - The *HTTP post URL* which we can push the message to you.
    - **preferLanguage** you want, it will affect the value within the pushing message, only support **en** and **zh_cn** at this moment.

2. After that, we will assign the unique ID for you and generate 40 characters long *API Token* to you. From now on, we will push the real-time message to you if it comes from any of your hardware devices.

3. Also, you can send us an *HTTP post request* as the command you want to send down to the specified hardware device.

Receiving the push message

Here is all the information you need to know for receiving the push message:

- You should set up an HTTP server for receiving the push message. Due to the reasons for protecting our platform, we ***only send once for each push message***, no error retry at this moment. We'll keep improving and add more robust and fault-tolerant handling in the future. However, at this moment, you better to make sure your server keeps running well for avoiding the missing message.

- Inside the message body of a single HTTP post request, all the device responses are *under time order*. However, we can't guarantee that the order of all device responses which come from different HTTP post request is *under the correct time order*, as the **TGP** is running in a high-performance clustering model. That's why you should take responsibility for sorting all the device responses you received either by the ***gpsLocation.recvTime*** or the ***elockResponse.time***. If both time value exists, please pick the ***gpsLocation.recvTime*** to go sorting, as that's the server time.

- We always add the custom HTTP header for each HTTP post to declare that the post request is from us.

```
{
    data-source-id: "TGP"
    data-type: "HDR" // For Hardware Device Response
}
```

- Below is the payload structure for each *HTTP post request* :

```
{
    // Device id as the key, and the device response array as the value
    "DEVICE_ID": [
        {
          "responseType": "Upload Data",
          "deviceId": "000019246001",
          "msgSeqNo": "2587",
          "elockResponse": {
            "cmdType": "Unseal Success",
            "elockId": "19016040",
            "gate": 4,
            "bill": "1907302029076087",
            "voltage": "4.0V",
            "status": "Unseal",
            "lineCode": "0",
            "unsealPassword": "547608",
            "cmdSource": "Platform",
            "time": "2019-08-01T12:52:27.000Z",
            "businessDataSeqNo": "7BFC",
          },
          "gpsLocation":
            { "alarm":
            "",
            "status": "Unseal,Precise Positioning,East Longitude,North Latitude",
            "isPrecise": true,
            "lat": "50.787781",
            "lon": "0.258630",
            "altitude": 4097,
            "speed": 0,
            "direction": 126,
            "gpsTime": "2019-08-01T12:52:26.000Z",
            "recvTime": "2019-08-01T12:52:31.431Z"
          },
          "extraInfoDescArr": [
            "LBS Info: Country Code - GB, Network identification - 15, Station LAC - 8208, Station CELL - 133441566, Signal strength - 20"
          ]
        }
    ],
    // Repeat for another device response
}
```

Attribute Name	*Description*
`DEVICE_ID`	*12 characters string, for example "000019246002"*
`responseType`	`"Upload Data"` *- ELock command response* `"Location Data Upload"` *- GPS location*
`deviceId`	*12 characters string, for example "000019246002"*
`msgSeqNo`	*Unique sequence number auto increased by hardware*
`elockResponse`	*This attribute only exists when* `responseType` *is* `Upload Data`*.*
`elockResponse.cmdType`	*Command response type, available string values:* *`Lock Info Upload`* *`Knob Broken Alarm`* *`Broken Alarm`* *`Cut Alarm`* *`Opened Alarm`* *`Apply For Dynamic Password / Apply For Unsealing`* *`Set Dynamic Password Success`* *`Modify Local Password Success`* *`Cancel Alarm Success`* *`Cancel Alarm Timeout`* *`Not In Alarm Status`* *`Key Not Match, Cancel Alarm Failure`* *`Shell Broken, Cancel Alarm Failure`* *`Emergency Unlocking, Cancel Alarm Failure`* *`Seal Success`* *`Repeated Sealing`* *`Not Locked, Seal Failure`* *`Low Voltage, Seal Failure`* *`Shell Broken, Seal Failure`* *`Emergency Unlocking, Seal Failure`* *`Cut Alarm, Seal Failure`* *`Opened Alarm, Seal Failure`* *`Seal Timeout`* *`Unseal Success`* *`Repeated Unsealing`* *`Opened, Unseal Failure`* *`Key Not Match, Unseal Failure`* *`Shell Broken, Unseal Failure`* *`Emergency Unlocking, Unseal Failure`* *`Opened Alarm, Unseal Failure`* *`Not Sealed, Unseal Failure`* *`Cut Alarm, Unseal Failure`* *`Unseal Timeout`*
`elockResponse.elockId`	*8 characters string, for example "19016040"*

`elockResponse.gate`	*Number*
`elockResponse.bill`	*16 characters string based on the time, for example "1907302029076087"*
`elockResponse.voltage`	*String, for example "4.0 V"*
`elockResponse.status`	*String, the ELock status, separated by comma for multiple status*
`elockResponse.linceCode`	*Number*
`elockResponse.sealPassword`	*6 characters string, this attribute only exists when* `elockResponse.cmdType` *is* `Seal Success`*.*
`elockResponse.unsealPassword`	*6 characters string, this attribute only exists when* `elockResponse.cmdType` *is* `Unseal Success`*.*
`elockResponse.cmdSource`	*Command source type, available string values:* *SMS* *Automatic* *Keyboard* *Hand Device* *Platform* *Gate* *IC Card* *Other*
`elockResponse.time`	*String, hardware time to run the command in UTC format*
`elockResponse.businessDataSeqNo`	*String, it will be the same HEX (4 ASCII Char) value with the* `businessDataSeqNo` *you sent through the command. If you don't set the value when you send a command, then the return value here will be a random HEX value.* *Or it will be* `0000` *when the response is a hardware automatic upload (not triggered by any command you send).* ***This attribute only available fo*** `TT808` (`"protocolType": "808"`) ***hardware device!!!***
`gpsLocation`	*This attribute always be there*
`gpsLocation.alarm`	*String, separated by comma for multiple alarm*
`gpsLocation.status`	*String, separated by comma for multiple status*
`gpsLocation.isPrecise`	*Boolean, indicate the GPS location is precise or not*
`gpsLocation.useLbsLocation`	*Boolean, set to* ***true****, that means* `gpsLocation.lat` *and* `gpsLocation.lon` *both use LBS location and the address will have the "(accuracy XXX meters)" description at the end.*

`gpsLocation.lat`	*Number, latitude*
`gpsLocation.lon`	*Number, longitude*
`gpsLocation.altitude`	*Number, altitude*
`gpsLocation.speed`	*Number, unit in km/h*
`gpsLocation.direction`	*Number, 0 ~ 360 degree*
`gpsLocation.time`	*String, hardware time in UTC format*
`gpsLocation.recvTime`	*String, server received time in UTC format*
`extraInfoDescArr`	*String array, the extra information*

Here is the real example:

```
{
  "000019246001": [
    // GPS location
    {
      "responseType": "Location Data Upload",
      "deviceId": "000019246001",
      "msgSeqNo": "0002",
      "gpsLocation": {
        "alarm": "Device Main Power Undervoltage",
        "status": "Standby,Precise Positioning,East Longitude,North Latitude",
        "isPrecise": true,
        "lat": "50.787776",
        "lon": "0.258606",
        "altitude": 4097,
        "speed": 0,
        "direction": 320,
        "gpsTime": "2019-08-06T06:40:19.000Z",
        "recvTime": "2019-08-06T06:40:20.193Z"
      },
      "extraInfoDescArr":
        [ "Odometer: 0
        KM",
        "Wireless Network Signal Strength: 7",
        "LBS Info: Country Code - GB, Network identification - 20, Station LAC -
5144, Station CELL - 929544, Signal strength - 17",
        "Device Power: 20%",
        "24 Alarm Status: Lock Rope Cut Alarm,Tilt Alarm",
        "24 Switch Status: Network Type - 4G,Vibration,Motor Released"
      ]
    },
    // ELock command response with GPS location
    {
      "responseType": "Upload Data",
      "deviceId": "000019246001",
```

```
      "elockResponse": {
        "cmdType": "Unseal Success",
        "elockId": "19246001",
        "gate": 4,
        "bill": "1908060640196087",
        "voltage": "4.0V",
        "status": "Unseal",
        "lineCode": "0",
        "unsealPassword": "547608",
        "cmdSource": "Platform",
        "time": "2019-08-06T06:40:19.123Z",
        "businessDataSeqNo": "68AE"
      },
      "gpsLocation":
        { "alarm": "",
        "status": "Unseal,Precise Positioning,East Longitude,North Latitude",
        "isPrecise": true,
        "lat": "50.787781",
        "lon": "0.258630",
        "altitude": 4097,
        "speed": 0,
        "direction": 126,
        "gpsTime": "2019-08-06T06:40:19.123Z",
        "recvTime": "2019-08-06T06:40:20.456Z"
      },
      "extraInfoDescArr": [
        "LBS Info: Country Code - GB, Network identification - 15, Station LAC -
8208, Station CELL - 133441566, Signal strength - 20"
      ]
    }
  ],

  "000019246002": [
    // GPS location with LBS location sample
    {
      "responseType": "Location Data Upload",
      "deviceId": "000019246002",
      "msgSeqNo": "0008",
      "gpsLocation": {
        "alarm": "Device Main Power Undervoltage",
        "status": "Standby,Precise Positioning,East Longitude,North Latitude",
        "isPrecise": false,
        "lat": "50.787776",
        "lon": "0.258606",
        "altitude": 4097,
        "speed": 0,
        "direction": 320,
        "gpsTime": "2019-08-06T06:40:19.000Z",
        "recvTime": "2019-08-06T06:40:20.193Z",
        "useLbsLocation": true,
        "address": "288 Xiangyun 1st Rd, Huli Qu, Xiamen Shi, Fujian Sheng, China,
```

```
360000 (accuracy 550 meters)"
,
      },
      "extraInfoDescArr":
        [ "Odometer: 0
        KM",
        "Wireless Network Signal Strength: 7",
        "LBS Info: Country Code - GB, Network identification - 20, Station LAC -
5144, Station CELL - 929544, Signal strength - 17",
        "Device Power: 20%",
        "24 Alarm Status: Lock Rope Cut Alarm,Tilt Alarm",
        "24 Switch Status: Network Type - 4G,Vibration,Motor Released"
      ]
    }
  ]
```

- And below are the **zh_cn** samples:

```
{
    "000000000001": [
        {
            "protocolType": "808",
            "responseType": "定位上传",
            "deviceId": "000000000001",
            "msgSeqNo":"000A",
            "gpsLocation": {
                "alarm": "终端主电源欠压",
                "status": "待命态,北斗,GLONASS,精确定位,东经,北纬",
                "isPrecise": true,
                "lat":"22.528786",
                "lon": "114.051955",
                "altitude": 22353,
                "speed": 0,
                "direction": 0,
                "gpsTime": "2019-01-31T11:21:56.000Z",
                "recvTime": "2020-02-27T04:28:01.072Z",
                "address": "广东省深圳市福田区福田街道皇庭世纪香江家居MALL(深圳福田店)"
            },
            "extraInfoDescArr":
                [ "里程: 0 KM",
                "无线网络信号强度: 29",
                "基站定位信息: 国家代码 - CN,网络标识 - 0,基站 LAC - 10129,基站 CELL
- 54317462,信号强度 - 29",
                "设备电压: 20%, 3500mV",
                "设备电量: 25%",
                "25路报警状态: 低温报警,高温报警",
                "24路开关状态: 网络类型 - 2G,振动,锁绳断开,开盖,拆壳"
            ]
        }
```

```
    ],

    // GPS location with LBS location sample
    "000000000002": [
        {
            "protocolType": "808",
            "responseType": "数据透传",
            "deviceId": "000000000002",
            "msgSeqNo":"6747",
            "elockResponse":{
                "cmdType": "施封成功",
                "elockId": "80018002",
                "gate": 4,
                "bill": "9111307445934210",
                "voltage": "3.8 V",
                "status": "施封态",
                "lineCode": "0",
                "sealPassword": "099342",
                "cmdSource": "中心",
                "time": "2019-11-13T07:45:00.000Z",
                "businessDataSeqNo": "68AE"
            },
            "gpsLocation":
                { "alarm":
                "",
                "status": "施封态,东经,北纬",
                "isPrecise":false,
                "lat":"23.020550",
                "lon": "113.306243",
                "altitude": 1,
                "speed": 3.7,
                "direction": 243,
                "gpsTime": "2019-11-13T07:44:58.000Z",
                "recvTime": "2020-02-27T04:28:03.072Z",
                "useLbsLocation": true,
                "address": "福建省厦门市湖里区禾山街道埭辽水库埭辽公园（精度 550 米）"
            },
            "extraInfoDescArr": [
                "基站定位信息: 国家代码 - CN,网络标识 - 0,基站 LAC - 0,基站 CELL - 82"
            ]
        }
    ]
```

- We don't expect that you'll send back an HTTP response result, we treat the **200 status** as succeed signal and ignore any return value.

Sending the command message

Here is all the information you need to know for sending the command message:

- We don't handle the off-line device case. If you try to send the command to any off-line device, we'll return fail and error message immediately.

- You should attach the following header in the *HTTP post request*, and use your token as the value. Otherwise, you'll get back a **401 UnauthorizedError**.

```
{
    Content-Type: "application/json"
    Authorization: "VB25taGElVs7SrFySdv14Or8IsZdO261QF5sxw8tW4IdVeWPFOhffA=="
}
```

- You need to send an *HTTP post request* to *https://api.totarget.net:8108/api/send-command* with the following payload structure:

```
{
    // Fixed "commands" key
    //Is Offline Cache delivery (Optional)?. This parameter is not
    delivered offline cache by default
    "cacheCommandsWhenOffline": false,
    "commands": {
        // Device id as the key, and the command array as the value
        // 12 characters string, for example: "000019246002"
        "DEVICE_ID": [
            {
                // Command type
                "type": "",
                // Command json payload here
            }
        ],
        // Repeat the commands for another device
    }
```

- Device setting command sample for **TT808** device:

```
// Device heartbeat interval
{
    "cacheCommandsWhenOffline": false,
    "commands": {
        "000019246002": [{
            "type": "ParameterSettings",
            "paramSettingList": [{
                "commandId": "00000001",
                // Unit in seconds, here set to 10 mins
                "heartbeatInterval": 600
            }]
        }],
    }
}

// Default location upload interval
{
    "cacheCommandsWhenOffline": false,
    "commands": {
        "000019246002": [{
            "type": "ParameterSettings",
            "paramSettingList": [{
                "commandId": "00000029",
                // Unit in seconds
                "defaultLocationUploadInterval": 30
            }]
        }],
    }
}

// Location upload interval in alarm mode
{
    "cacheCommandsWhenOffline": false,
    "commands": {
        "000019246002": [{
            "type": "ParameterSettings",
            "paramSettingList": [{
                "commandId": "00000028",
                // Unit in seconds
                "inAlarmLocationUploadInterval": 30
            }]
        }],
    }
}

// Location upload interval in sleep mode
{
    "cacheCommandsWhenOffline": false,
    "commands": {
        "000019246002": [{
            "type": "ParameterSettings",
            "paramSettingList": [{
                "commandId": "00000027",
                // Unit in seconds
                "sleepingLocationUploadInterval": 30
            }]
        }],
    }
}
```

```
// device PowerOff
{
  "cacheCommandsWhenOffline":false,
  "commands":{
    "000019246002":[{
        "type":"DevicePowerOff"
      }]
  }
}

// Reset Factory Status
{
  "cacheCommandsWhenOffline":false,
  "commands":{
    "000019246002":[{
        "type":"DeviceFactoryReset"
      }]
  }
}

// Reboot Device
{
  "cacheCommandsWhenOffline":false,
  "commands":{
    "000019246002":[{
        "type":"RebootDevice"
      }]
  }
}
// Reset Defaullt IP
{
  "cacheCommandsWhenOffline":false,
  "commands":{
    "000019246002":[{
                type: "UrgentContentToDevice"
                content: "SIP0"
      }]
  }
}

//Enable Sleep Mode
{
  "cacheCommandsWhenOffline":false,
  "commands":{
    "000019246002":[{
                type: "EnableSleepMode"
      }]
  }
}
```

- ELock command sample for **TT808** device:

```
// Seal
{
    "commands": {
        "000019246002": [{
            // At this moment, we ONLY support single ELock command per sending
            "type": "Elock",
            "elockCommand": {
                "cmdType": "SEAL",
                "lockId": "19246002",
                "bill": "1908060640196086",
                "lineCode": 1234,
                "gate": 8,
                "key": "13970",
                "validTime": 10,
                "businessDataSeqNo": "A1B2"
            }
        }]
    }
}
```

Attribute Name	*Description*
`cmdType`	*ELock command type, available string values:* *SEAL* *UNSEAL* *CANCEL_ALARM* *MODIFY_LOCAL_PASSWORD* *SET_DYNAMIC_PASSWORD*
`lockId`	*8 characters string, for example "19016040"*
`bill`	*16 characters string, we recommend that generate the value based on the current time to avoid the duplicate case. For example "1907302029076087"*
`linceCode`	*Number, 4 digital, 0000 ~ 9999*
`gate`	*Number, 1 ~ 255*
`key`	*String, 6 digit number as the password*
`validTime`	*When* `cmdType` *is* *SET_DYNAMIC_PASSWORD* *or* *MODIFY_LOCAL_PASSWORD*, *this attribute to set the valid time for the dynamic password. Otherwise, it will be ignored.* *0 ~ 255, unit in minute. If set to 0, that means clear the*

	dynamic password.
`businessDataSeqNo`	*String, 4 ASCII characters, each character can be any one of: **'0' ~ '9'**, **'a' ~ 'f'**, **'A' ~ 'F'**.* *The ELock Response will send back exactly the same value in `elockResponse.businessDataSeqNo`, then you can use it to figure out that response is related to which command you sent.* ***This parameter only available for `TT808` (`"protocolType": "808"`) hardware device!!!***

Here is the **TT808** device real example you can copy & paste into the Post-man

```
{
    "commands": {
        "000019246001": [
            {
                "type": "Elock",
                "elockCommand": {
                    "cmdType": "SEAL",
                    "lockId": "19246001",
                    "bill": "1908060640196081",
                    "lineCode": 1234,
                    "gate": 8,
                    "key": "13970",
                    "validTime": 0,
                    "businessDataSeqNo": "712E"
                }
            }
        ]
    }
}
```

Here is the **TT808** device real example you can send via **curl**:

```
curl -X POST https://api.totarget.net:8108/api/send-command \
--header 'Content-Type: application/json' \
--header 'Authorization': 'VB25taGElVs7SrFySdv14Or8IsZdO261QF5sxw8tW4IdVeWPFOhffA==' \
--data '{
    "commands": {
        "000019246001": [
            {
                "type": "Elock",
                "elockCommand": {
                    "cmdType": "SEAL",
```

```
                "lockId": "19246001",
                "bill": "1908060640196081",
                "lineCode": 1234,
                "gate": 8,
                "key": "13970",
                "validTime": 0,
                "businessDataSeqNo": "712E"
            }
        }
    ]
  }
}'
```

- HTTP response
  a. Error
     - If you don't provide the `Authorization` header with your valid token value, you'll get back a **401 UnauthorizedError**.

     - If you try to send the command to any hardware device which not belongs to you, you'll get back a **401 UnauthorizedError** with the following error message:

       ```
       One of the "device id" not belongs to current TIS.
       ```

     - If you don't provide the valid post body, you'll get back a **400 BadRequestError** with the following error message:

       ```
       {
          "code": "BadRequest",
          "message": "The payload must have a valid structure like
       \"{ commands: { deviceId: [{ type: string }] } }\""
       }
       ```

  b. Success
     - If you try to send the command to any offline device, you'll get back the result like below:

       ```
       [
          {
              "000019246001": [
                  {
                      "commandDesc": "[ Elock ]",
                      "success": false,
       ```

```
            "errorMessage": "Device is offline"
        }
    ]
},
{
    "000019256002": [
        {
            "commandDesc":"[Elock]",
            "success": false,
            "errorMessage": "Device is offline"
        }
    ]
}
]
```

- If all commands had been handled well, you'll get back the success result like below:

```
[
    {
        "000019246001": [
            {
                "commandDesc": "[ Send Data ]\nCommand Type:
Seal\nElock Id: 19246001\nBill: 1908060640196081\nLine Code:
1234\nGate: 8\nKey: 168168\nValid Time: 0 Minutes\n",
                "success": true
            }
        ]
    },
    {
        "000019256002": [
            {
                "commandDesc": "[ Send Data ]\nCommand Type:
Cancel Alarm\nElock Id: 19256002\nBill: 1908060640196089\nLine
Code: 5678\nGate: 3\nKey: 13970\nValid Time: 0 Minutes\n",
                "success": true
            }
        ]
    }
]
```

- Of course, we can deal with the online and offline case:

```
[
    {
        "000019256002": [
            {
                "commandDesc": "[ Elock ]",
                "success": false,
                "errorMessage": "device offline"
```

```
            }
        ]
    },
    {
        "000019246001": [
            {
                "commandDesc": "[Send Data]\nCommand Type:
Seal\nElock Id: 19246001\nBill: 1908060640196081\nLine Code:
1234\nGate: 8\nKey: 168168\nValid Time: 0 Minutes\n",
                "success": true
            }
        ]
    }
]
```

- Device command sample for **TTKJ** device:

```
// Single location upload
{
    "commands": {
        "000000000001": [{
            "type": "SingleReportLocation"
        }]
    }
}

// Default location upload interval
{
    "commands": {
        "000000000001": [{
            "type": "ReportLocation",
             // Unit in seconds
            "interval": 300
        }]
    }
}

// Rest factory settings
{
    "commands": {
        "000000000001": [{
            "type": "ResetFactorySettings"
        }]
    }
}

// Reboot device
{
    "commands": {
        "000000000001": [{
            "type": "RebootDevice"
        }]
```

```
    }
}

// Restore factory settings (IP)
{
    "commands": {
        "000000000001": [{
            "type": "RestoreFactorySettings(IP)"
        }]
    }
}




// Reboot GPS module
{
    "commands": {
        "000000000001": [{
            "type": "RebootGPS"
        }]
    }
}

// ELock command
{
    "commands": {
        "000000000001": [{
            "type": "Elock",
            "elockCommand": {
                "cmdType": "SEAL",
                "lockId": "123456",
                "bill": "1122AABBCCDD",
                "lineCode": 1234,
                "gate": 8,
                "key": "13970",
                "validTime": 10
            }
        }]
    }
}
```