Data Telemetry from BLE Devices

Registering Topics

Register Topics - API Definition

API	Description
`/control/registration/registerTopic`	This API registers the topic with the gateway to subscribe to GATT notifications from device using ID, service UUID, and characteristic UUID. The subscriptions can be placed to various GATT attributes: • Blood Oxygen level • Heart rate • Walking rate

API Example - Register Topics

Register Topics Request:

```
{
"technology": "ble",
"ids": [
"df553860-975c-4d7f-8c1f-b2996f34e26f",
],
"controlApp": "controlApplication",
"topic": "enterprise/hospital/pulse_oximeter",
"dataFormat": "default",
"ble": {
```

```
"type": "gatt",
"serviceID": "1800",
"characteristicID": "2A5E"
}
}
```

Register Topics Response:

```
{
    "status": "SUCCESS",
    "topic": "enterprise/hospital/pulse_oximeter",
    "reason": "Gatt topics successfully registered!"
}
```

API Example - Subscribe to Critical Application Events

This subscription helps identify if the IoT Orchestrator has experienced a critical event (for example, upgrade process, uninstallation, or restart). In the current IoT Orchestrator implementation, only devices in the Onboarded state are persisted in the database. Devices in any state other than Onboarded will not be persisted; therefore, if a critical event occurs in the IoT Orchestrator, a new onboarding process is required. This API provides information about these critical events for a particular topic.

Critical Request Subscription Request:

```
{
    "technology": "ble",
    "controlApp": "controlApplication",
    "topic": "enterprise/hospital/pulse_oximeter",
    "ble": {
        "type": "application_events"
}
}
```

Critical Request Subscription Response:

```
{
  "status" : "SUCCESS",
  "topic" : "enterprise/hospital/pulse_oximeter",
  "reason" : "Application event topics successfully registered!"
}
```

Telemetry Data Format

For information, see https://github.com/iot-onboarding/non-ip-iot-control/blob/nipc-asdf-01/proto/data_app.proto.

Telemetry Message Format

The telemetry messages are in **protobuf** format. All the telemetry messages are encapsulated in the **DataSubscription** message.

Subscribing to Advertisements and Notifications

Subscribing to Advertisements and Notifications - API Definition

API	Description
/control/data/subscribe	Use this API to enable GATT notifications or indications in the BLE device using the device ID, GATT service UUID, and characteristic UUID.

API Example – Subscribing to Advertisements and Notifications

Request:

```
{
"technology": "ble",
"id": "df553860-975c-4d7f-8c1f-b2996f34e26f",
"ble": {
"serviceID": "1800",
"characteristicID": "2A5E"
},
"controlApp": "controlApplication"
}
```

Response:

```
{
"status": "SUCCESS",
"id": "df553860-975c-4d7f-8c1f-b2996f34e26f",
"requestID": "12345678-5678-1234-5578-abcdef1234"
}
```

MQTT Subscription Messages

Once the notification is enabled using the subscribe API, the data receiver application starts to receive the GATT notification. To receive the notification, you will need to execute the following command in your terminal session:

```
mosquitto_sub -h localhost -p 41883 -t
enterprise/hospital/pulse_oximeter -u
https://dataApplication --pw
c4e80e0483af0a4394dfb6e3ec5e820b
```

Telemetry Message Format for Onboarded Advertisements and Notifications

The following is the code snippet of Telemetry message format:

```
syntax = "proto3";
```

```
import "google/protobuf/timestamp.proto";

option java_package = "org.ietf.nipc.proto";
option java_multiple_files = true;

package nipc;

message DataSubscription {
    optional string device_id = 1;
    bytes data = 2;
    google.protobuf.Timestamp timestamp = 3;
    optional string ap_mac_address = 4;

    reserved 5 to 10;

    oneof subscription {
        BLESubscription ble_subscription = 11;
        BLEAdvertisement ble_advertisement = 12;
        ZigbeeSubscription zigbee_subscription = 13;
        RawPayload raw_payload = 14;
        BLEConnectionStatus ble_connection_status = 15;
        ApplicationEvent application_event = 16;
    }

    message BLESubscription {
        optional string service_uuid = 1;
        optional string characteristic_uuid = 2;
    }

    message BLEAdvertisement {
        string mac_address = 1;
        optional int32 rssi = 2;
    }

    message ZigbeeSubscription {
        optional int32 endpoint_id = 1;
        optional int32 cluster_id = 2;
        optional int32 attribute_id = 3;
        optional int32 attribute_type = 4;
    }

    message BLEConnectionStatus {
        string mac_address = 1;
        bool connected = 2;
        optional int32 reason = 3;
    }

    message RawPayload {
        optional string context_id = 1;
    }

    message ApplicationEvent {
        string message = 1;
    }
}

message DataBatch {
    repeated DataSubscription messages = 1;
}
```

Subscribing to Connection Events

Subscribing to Connection Events - API Definition

API	Description
/control/registration/registerTopic	Use this API to register a topic to provide notifications for device connection and disconnection events.

API Example – Subscribing to Connection Events

Request:

```
{
    "technology": "ble",
    "ids": [
        "df553860-975c-4d7f-8c1f-b2996f34e26f","
    ],
    "controlApp": "controlApplication",
    "topic": "enterprise/hospital/conn_events",
    "dataFormat": "default",
    "ble": {
        "type": "connection_events"
    }
}
```

Response:

```
{
  "status" : "SUCCESS",
  "topic" : "enterprise/hospital/conn_events",
  "reason" : "Successfully registered Connection event topics.",
  "registeredIDs" : [ "df553860-975c-4d7f-8c1f-b2996f34e26f" ]
}
```