＜使い方　プッシュ受信＞

Kii Cloud では通常のデータ送信では REST API を使い、プッシュ通知には MQTT で実現しています。Kii Cloud のオンラインドキュメントによると、MQTT でのプッシュ受信に関しては任意の MQTT ライブラリを使うようにと言ってますので、今回はオープンソースの MQTT Client「mosquitto」を使います。ちなみに、Kii が提供する MQTT ブローカーは Kii クラウドでのプッシュ通知専用になりますので、サブスクライバーに大して、任意のデータを直接パブリッシュすることは出来ませんので注意して下さい。Kii でプッシュを受けるには、Kii Cloud のスコープやトピック（MQTT のトピックと、KiiCloud 内のトピックは違う）、Bucket などと言った用語の意味を理解し、プッシュ受信の設定をする必要がありますが、今回は、先ほどデータ送信の例で作成した「test」という Backet へ Thing スコープで送信した場合、プッシュを受ける例を紹介します。

１．Thing のインストール

Kii Cloud へ今回のプッシュ受信対象の Thing をインストールします。こちらも REST API で行います。

```
$ curl -v -X POST ¥
-H "Authorization: Bearer {ACCESS_TOKEN}" ¥
-H "X-Kii-AppID: {APP_ID }" ¥
-H "X-Kii-AppKey: {APP_KEY}" ¥
-H "Content-Type: application/vnd.kii.InstallationCreationRequest+json" ¥
"https://api-jp.kii.com/api/apps/{APP_ID }/installations" ¥
-d '{
            "deviceType": "MQTT",
            "development": true
}'

以下はレスポンス結果
< HTTP/1.1 201 Created
{
   "installationID" : "[redacted]",
   "installationRegistrationID" : "[redacted]"
}
```

以下のコマンドで確認します。登録時のレスポンスで得た installationID を使います。

```
$ curl -v -X GET ¥
-H "Authorization: Bearer {ACCESS_TOKEN}" ¥
-H "X-Kii-AppID: {APP_ID }" ¥
-H "X-Kii-AppKey: {APP_KEY}" ¥
"https://api-jp.kii.com/api/apps/{APP_ID }/installations/{installationID}"

以下はレスポンス結果
< HTTP/1.1 200 OK
{
            "installationID" : "[redacted],
            "installationRegistrationID" : "[redacted]-ad91--9a82-[redacted],
            "development" : true,
            "thingID" : "th.[redacted]c21a-[redacted]-c385-",
            "deviceType" : "MQTT"
```

```
}
```

２．Buckets の購読設定

どの Bucket が更新されたらプッシュを受けるか？の設定を REST API で行います。{} の部分は環境に合わせて変更して下さい。

{ACCESS_TOKEN}：データ送信の際に作成された Access Token を指定します

{APP_ID} {APP_KEY}：ダッシュボードから確認出来ます。

{Things_ID}：データ送信の際に作成された Things_ID を指定します。

```
$ curl -v -X PUT ¥
-H "Authorization: Bearer {ACCESS_TOKEN}" ¥
-H "X-Kii-AppID: {APP_ID}" ¥
-H "X-Kii-AppKey: {APP_KEY}" ¥
"https://api-jp.kii.com/api/apps/{APP_KEY}/things/{Things_ID}/buckets/test/filters/all/push/subscriptions/things/{Things_ID}" ¥
-d ""
```

作成後、以下のコマンドで確認します。

```
$ curl -v -X GET ¥
-H "Authorization: Bearer {ACCESS_TOKEN}" ¥
-H "X-Kii-AppID: {APP_ID }" ¥
-H "X-Kii-AppKey: {APP_KEY}" ¥
"https://api-jp.kii.com/api/apps/{APP_ID}/buckets/test/filters/all/push/subscriptions/things/{Things_ID}"
```

３．MQTT エンドポイントの取得

MQTT のエンドポイント（接続情報）を取得します。なお、このコマンドはサーバ側の準備が出来ていない場合、503 を返す場合があります。その場合は何度かリトライして下さい。

```
$ curl -v -X GET ¥
-H "Authorization: Bearer {ACCESS_TOKEN}" ¥
-H "X-Kii-AppID: {APP_ID}" ¥
-H "X-Kii-AppKey: {APP_KEY}" ¥
"https://api-jp.kii.com/api/apps/{APP_ID}/installations/{installationID}/mqtt-endpoint"

以下はレスポンス結果
< HTTP/1.1 200 OK
{
   "installationID" : "{installationID}",
   "username" : "0faa3125-[redacted]",
   "password" :"ii[redacted]Ys",
   "mqttTopic" : "9[redacted]8",
   "host" : "jp-mqtt-[redacted].kii.com",
   "portTCP" : 1883,
   "portSSL" : 8883,
   "X-MQTT-TTL" : 2147483647
}
```

４．MQTT でのサブスクライブ

ようやく MQTT でデータを受信する準備が整いましたので、mosquitto_sub でサブスクライブします。以下のコマンドを実行し受信待ち状態を作ります。オプションで入れるデータは先ほど得た MQTT エンドポイントを参照して下さい。

```
$ mosquitto_sub -d -h {host} -u {username} -t "{mqttTopic}" -P {password} -i
"{mqttTopic}"
```

５．プッシュ受信

Bucket 名「test」へデータ投入するとサブスクライブされます。先ほどの mosquitto_sub コマンドとは別のターミナルを開きデータ投入を行います。

```
$ curl -v -X POST ¥
-H "Authorization: Bearer {ACCESS_TOKEN}" ¥
-H "X-Kii-AppID: {APP_ID}" ¥
-H "X-Kii-AppKey: {APP_KEY}" ¥
-H "Content-Type: application/vnd.{APP_ID}.mydata+json" ¥
"https://api-jp.kii.com/api/apps/{APP_ID}/things/{THING_ID}/buckets/test/objects" ¥
-d '{"score": "1800", "name": "game1"}'
```

コマンド投入後、サブスクライバーがプッシュ受信する様子が確認出来ます。