ZIMI API Beta

Revision History

Revision	Change	Name	Date
0.1	Document Created	J.Bentley	1/5/2021

ZCC API Documentation

- Overview
- Terminology
- Components
  - ZCC
  - ZCC UDP Discovery
  - ZCC TCP Socket Connection
- Prerequisites
- API Calls
  - ZCC API Requests
    - Get ControlPoint Properties
    - GET ControlPoint States
    - GET Devices Actions
    - Apply ControlPoint Actions
  - Cloud API Events
    - Subscribe to ControlPoint status updates
    - Unsubscribe to ControlPoint status updates

Overview

ZCC API is a TCP socket based api that communicates using json messages.

Terminology

Device

A Zimi device is a physical electrical component that can have 1 or many (up-to 6) “control points”. Devices are (Dimmer, Dual GPO, 1g Multi purpose switch, Fan Controller, Multi purpose switch (1g - 4g), Garage Controller

ControlPoint

A individually controllable output on a Device.

Dual GPO 2 x Outlets

Dimmer 1 x Dimmable switch

Multi purpose Switch 1 - 4 x Switches

Fan controller 1 x fan , 1x switch

Garage controller 2x doors

ControlPoint Properties

Attributes of the device that identify the device individually. These can be id, macAddress, controlpoint type, name, roomName

ControlPoint States

States of the control point. Based on the ControlPoint type, these can be whether it is turned on or off, brightness, speed, opened, closed.

ControlPoint Actions

The types of control change actions allowed by the ControlPoint. These can be TurnOn, TurnOff, SetBrightness, OpenDoor, CloseDoor.

Components

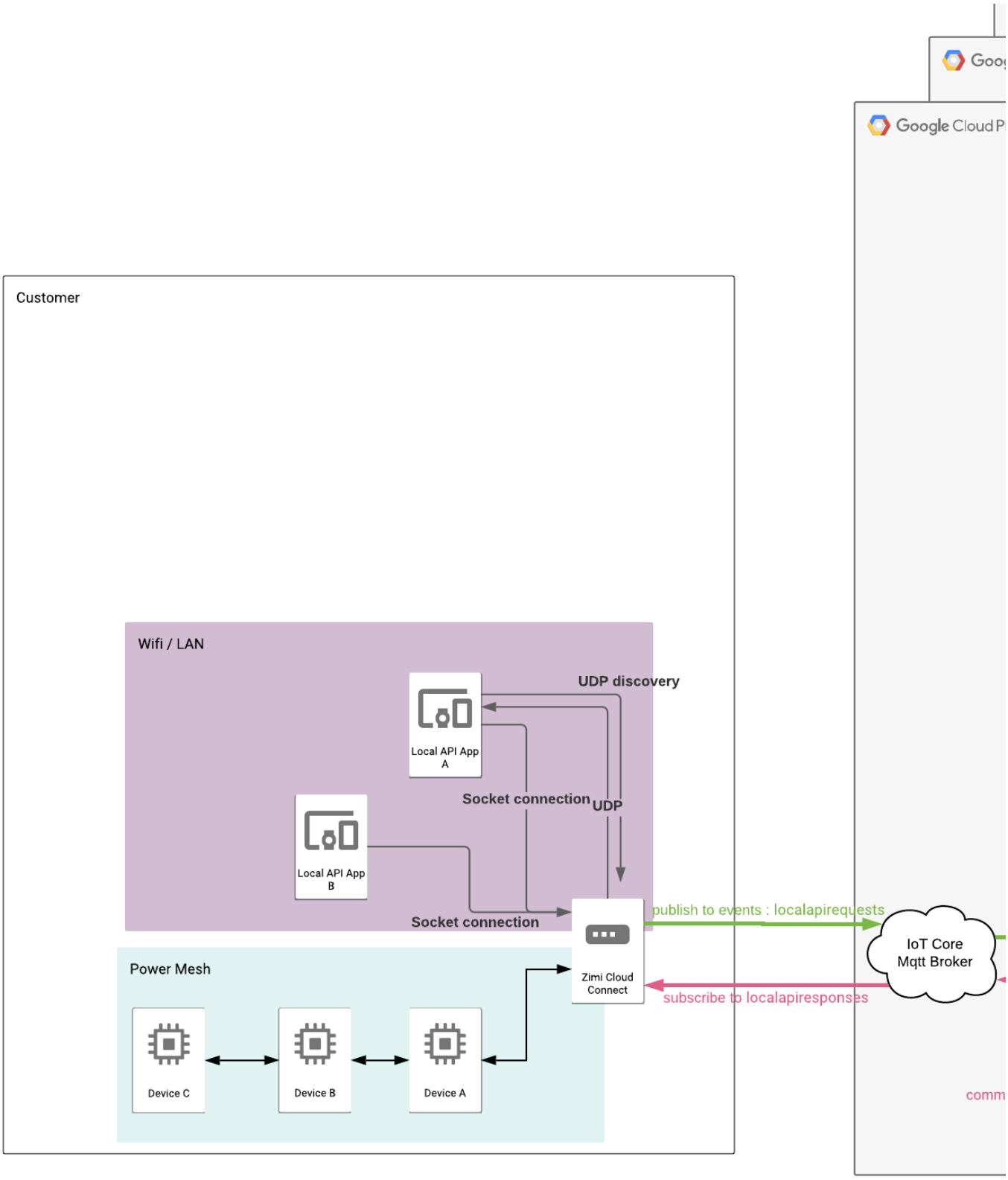
ZCC

Zimi cloud connect (ZCC) connects Zimi devices via ble and acts as a gateway to the internet. Zimi Devices can be controlled remotely via the internet using the ZCC.

ZCC API will be available through a TCP server available on the ZCC. It can be accessed by connecting to the same wifi network and making a TCP socket connection with the ZCC

Zimi Devices <--*ble*--> **ZCC** <--- *wifi*---> **WifiRouter** <--- *internet* ---> **Zimi Cloud**

ZCC(TCP Socket) <--- *wifi* ---> **API App**



ZCC UDP Discovery

ZCC can be discovered on the wifi network using UDP discovery

Client broadcasts UDP message

Attribute	Description	Value
Protocol	Internet protocol	UDP
Address	Address for broadcasting message on UDP.	255.255.255.255
Port	Destination port for the UDP broadcast.	5001
Packet	Payload to send in the UDP broadcast. ZCC responds only if client broadcasts correct packet.	ZIMI

Client receives ZCC's response

Attribute	Description	Value
Protocol	Internet protocol	UDP
Port	Listen port for the UDP broadcast.	5002
Packet	Message from ZCC	Example: `{ "brand": "zimi", "product": "zcc", "mac": "c4ffbc90abde", "tcp": 5003, "availableTcps": 6 }` Items' Description: **brand**: Brand of device **product**: Type of product **mac**: MAC address of device **tcp**: Available TCP port **availableTcps**: Show the number of available TCP clients

ZCC TCP Socket Connection

Start local API by connecting to ZCC on TCP.

Attribute	Description	Value
Protocol	Internet protocol	TCP
Address	Address of ZCC	X.X.X.X (Connect to address that responds message in the UDP discovery)
Port	TCP port for local API	5003 (Depends on the message from UDP discovery)
Packet	Payload to communicate with ZCC	… (Please read the description of local API)

Prerequisites

In order to use the ZCC api, you need to have a Zimi device network setup with multiple devices and a ZCC device.

- Use the Zimi app on Google Play store or Apple IOS App store
    - Create a Zimi user account
- Create a Zimi network with devices
    - Using Zimi app, Create a network
    - Using Zimi app, Commission devices to the network
    - Using Zimi app, Commission ZCC device to the network
- Request ZCC api enabled ZCC firmware from Zimi support team
- Discover the ZCC's ip and port using discovery method described above
- Connect to the ZCC socket using the discovered ip and port.
- Send request messages as described below and listen for response messages and events.

API Calls

ZCC API Requests

Get ControlPoint Properties

Request to return all ControlPoints and their properties. Rate limited at maximum rate of 1 request every minute.

request message :

```
{
  "request": {
    "path": "api/v1/controlpoint/properties",
    "method": "GET"
  }
}
```

response message :

This can be sent as multiple batches of responses

```
{
  "response": {
    "controlpoint_properties": [
        {
            "id": "product0001_1",
            "properties": {
                "name": "kettle",
                "controlpointType": "outlet",
                "roomName": "kitchen"
                "roomId": 0
            }
        },
        {
            "id": "product0002_1",
            "properties": {
                "name": "dimmer light",
                "controlpointType": "dimmer",
                "roomName": "kitchen",
                "roomId": 1
            }
        }
      ]
  }
}
```

parameter	Possible values
controlpointType	• outlet • dimmer • light • fan • switch • garagedoor

GET ControlPoint States

Request to return all ControlPoint states.

request message :

```
{
  "request": {
    "path": "api/v1/controlpoint/states",
    "method": "GET"
  }
}
```

response message :

This can be sent as multiple batches of responses

```
{
  "response":{
      "controlpoint_states": [
          {
              "id": "product0001_1",
              "states": {
                  "controlState": {
                      "outlet": {
                          "isOn": true
                      }
                  },
                  "isConnected": true
              }
          },
          {
              "id": "product0002_1",
              "state": {
                  "controlState": {
                      "dimmer": {
                          "brightness": 50
                      }
                  }
                  "isConnected": true
              },
          }
        ]
  }
}
```

parameter	controlPointType	States
controlState	• outlet	`"outlet": { "isOn": boolean }`
	• dimmer	`"dimmer": { "isOn": boolean, "brightness": number // 0 - 100 }`

	• light	`"light": { "isOn": boolean }`
	• fan	`"fan": { "isOn": boolean "fanspeed": number // 0 - 7 }`
	• switch	`"switch": { "isOn": boolean }`
	• garagedoor	`"garagedoor": { "openpercentage": number // 0 - 100 }`

GET Devices Actions

request message :

```
{
  "request":{
    "path": "api/v1/controlpoint/actions"
    "method": "GET"
  }
}
```

response message :

This can be sent as multiple batches of responses

```
{
  "response":{
    "controlpoint_actions": [
```

```
{
    "id": "product0001_1",
    "controlpointType": "outlet"
    "actions": {
        "TurnOn": {
            "actionParams": {}
        },
        "TurnOff": {
            "actionParams": {}
        }
    }
},
{
    "id": "product0001_1",
    "controlpointType": "switch"
    "actions": {
        "TurnOn": {
            "actionParams": {}
        },
        "TurnOff": {
            "actionParams": {}
        }
    }
},
{
    "id": "product0002_1",
    "controlpointType": "dimmer"
    "actions": {
        "TurnOn": {
            "actionParams": {}
        },
        "TurnOff": {
            "actionParams": {}
        },
        "SetBrightness": {
            "actionParams": {
                "brightness": {
                    "type": "number",
                    "max": 100,
                    "min": 0
                }
            }
        },
        "Fade": {
            "actionParams": {
                "brightness": {
                    "type": "number",
                    "max": 100,
                    "min": 0
                },
```

```
                    "timeperiod": {
                        "type": "number",
                        "max": 100,
                        "min": 1
                    }
                }
            }
        }
    },
    {
        "id": "product0003_1",
        "controlpointType": "fan"
        "actions": {
            "TurnOn": {
                "actionParams": {}
            },
            "TurnOff": {
                "actionParams": {}
            },
            "SetFanSpeed": {
                "actionParams": {
                    "fanspeed": {
                        "type": "number",
                        "max": 7,
                        "min": 0
                    }
                }
            }
        }
    },
    {
        "id": "product0004_1",
        "controlpointType": "garagedoor"
        "actions": {
            "OpenDoor": {
                "actionParams": {}
            },
            "CloseDoor": {
                "actionParams": {}
            },
            "OpenToPercentage":{
                "actionParams": {
                  "openpercentage":
                    {
                        "type": "number",
                        "max": 100,
                        "min": 0
                    }
                }
            }
```

```
            }
        }
      ]
    }
}
```

Apply ControlPoint Actions

request message :

```
{
  "request": {
    "path": "api/v1/controlpoint/actions",
    "method": "POST",
    "body": {
      "actions":[
        {
          "id": {controlpointId},
          "action": "TurnOn"
        },
        {
          "id": {controlpointId},
          "action": "SetBrightness",
          "actionParams": {
            "brightness": 100
          }
        },
        {
          "id": {controlpointId},
          "action": "SetFanSpeed",
          "actionParams": {
            "fanspeed": 7
          }
        }
      ]
    }
  }
}
```

response message :

Response will not be sent if the action succeeded. A error response if be sent if the action could not be applied.

```
{
  "response": {
    "controlpoint_setactions": [
      {
        "id": {controlpointId},
        "action": "SetBrightness",
        "result": "fail",
        "error": "error details"
      },
      {
        "id": {controlpointId},
        "action": "SetFanSpeed",
        "result": "fail",
        "error": "error details"
      }
    ]
  }
}
```

Cloud API Events

Subscribe to ControlPoint status updates

request message :

```
{
  "request": {
    "path": "api/v1/subscribe/controlpoint/states",
    "method": "POST"
  }
}
```

event messages:

state structure is similar to get states contents.

```
{
  "response": {
    "controlpoint_states_events": [
        {
            "id": "product0001_1",
            "state": {
                "controlState": {
                    "outlet": {
                        "isOn": true
                    }
                },
                "isConnected": true
            }
        },
        {
            "id": "product0002_1",
            "state": {
                "controlState": {
                    "dimmer": {
                        "brightness": 50
                    }
                }
                "isConnected": true
            },
        }
      ]
  }
}
```


Unsubscribe to ControlPoint status updates

request message :


```
{
  "request": {
    "path": "api/v1/unsubscribe/controlpoint/states",
    "method": "POST"
  }
}
```