Accessing the Rako Hub from an external application.

Contents

1 Introduction

The Rako HUB can be accessed via a TCP socket or the legacy interfaces specified in 'Accessing The Rako Bridge'. The API is intended for controlling Rako products from an external system. HTTP commands are available via the Rako Bridge API. The WRA-232 should be used for sending outbound 232/TCP/UDP messages from Rako.

2 TCP

Products:

> *RK-HUB*
> *WK-HUB*

The TCP interface is available on port 9762. It can be used in BASIC or JSON mode. After the client connects, you must use the **SUB** message to select the protocol. Line termination can be the following: '\r', '\n' or '\r\n'. The encoding is UTF-8. Note: The legacy protocol uses port 9761 which is described in 'Accessing The Rako Bridge'.

Client to HUB commands are the following:

SUB: Must be sent as the first message to identify the client. Selects either BASIC or JSON.
STATUS: HUB identifiers and version info.
SEND: Transmit commands from the HUB.
QUERY: Query room data and levels info.

HUB to client feedback is provided in the following two ways:

Feedback: Triggered by "Button presses" may not result in a level change. Outputs all Rako commands.

Tracker: Occur when the level or scene of a dimmer changes. Only outputs level and scene changes.

3 BASIC

BASIC Commands (Client -> Hub)

[JSON Connection described later](#)

SUB

Name	SUB (BASIC)		
Parameters			
	Arg	**Name**	**Example**
	0	Command	SUB
	1	Format	BASIC
	2	Version	V3
	3	Client name	MY_DRIVER
	4	Subscriptions	TF
Description	Subscriptions: **T**: Tracker events **F**: Feedback events		
Request Example	SUB,BASIC,V3,MY_CONTROL_SYSTEM\r\n SUB,BASIC,V3,MY_CONTROL_SYSTEM,TF\r\n		
Example Description	Subscribe as MY_CONTROL_SYSTEM. Basic format Version 2 to Tracker and Feedback.		
Response Example	A,1		

STATUS

Name	STATUS (BASIC)		
Request			
	Arg	Name	Example
	0	Status	STATUS
	1	RFU	0
Request Example	STATUS,0\r\n		
Response			
	Arg	Name	Example
	0	Command	STATUS
	1	Product Type	Hub
	2	Protocol Version	V3
	3	HUB Id 36 character UUID	3ab4696c-8844-4ab9-a48c-2c95fcb5f0e3
	4	Mac	70:B3:D5:08:40:00
	5	HUB Version	3.0.6
	6	RFU	0
Response example	STATUS,Hub,V3,353e89dd-862c-3256-aa79-8c3b89c23f21,70:B3:D5:08:40:00,3.0.6,0\r\n		

SEND

Name	SEND (BASIC)		
Request			
	Arg	Name	Example
	0	Command	SEND
	1	Room	4
	2	Channel	1
	3	Command	SCENE (See commands table)
	4..7	Argument(s)	4
Request Example	SEND,4,1,SCENE,5\r\n SEND,1,2,LEVEL,3\r\n SEND,5,4,RGB,127,0,255\r\n		
Response			
	Arg	Name	Example
	0	Command	SEND
	1	Success	1
Response example	SEND,1 AERROR,UNKNOWN_ERROR		
Response Notes	The result '1' indicates the command has been successfully queued. It may fail to execute when processed. It is recommended to check the tracker feedback to ensure the command performed.		

BASIC Commands

Command	Description
SCENE	Scene 0-16 (0 = off)
LEVEL	Level (0-255)
FADE_UP	Start fading up
FADE_DOWN	Start fading down
FADE_STOP	Stop fading
STORE	Store currently selected scene / levels
RGB	Provide three 0-255 values for RGB. e.g. green = 0,255,0 White values are automatically calculated from the RGB

RGB example
RGB send/response example with RGBW strip on room 85 channel 6. Connection subscribed (SUB) with 'TF' flags.
| ```
SEND,85,6,RGB,150,255,255
F,85,8,LEVEL,123
F,85,6,LEVEL,210
F,85,7,LEVEL,210
F,85,9,LEVEL,255
SEND,1
T,85,8,LEV,150,123,168,0
T,85,6,LEV,255,210,280,0
T,85,7,LEV,0,210,1310,0
T,85,9,LEV,0,255,1591,0
``` |
| User sends the 'SEND' command.
Hub replies with 4 'F' feedback commands are sent with roughly 500ms delay.
Hub replies with SEND,1.
Hub replies with 4 'T' tracker messages showing current and target levels. |

QUERY

Strings with invalid characters for CSV will be escaped, when this happens the entire string will be quoted.

Name	QUERY (BASIC)		
Request			
	Arg	Name	Example
	0	Command	QUERY
	1	Type	ROOM
	2	Room Id (Optional. If missing uses room 0 (All rooms))	17
	3	Arguments (Optional)	N
	Arguments **N: No header**		
Response			
	Line	Name	Example
	0	Column headers	QUERY,ROOM_ID,ROOM_TITLE,ROOM_TYPE\r\n
	1..x	Data	17,Master Bedroom,LIGHT\r\n
Note	Type can be ROOM, CHANNEL or LEVEL. See tables below.		

Room Query

Request Example	`QUERY,ROOM\r\n`
	`QUERY,ROOM,0,N\r\n`
	`QUERY,CHANNEL,0\r\n`
	`QUERY,LEVEL,17,N\r\n`

Response example (ROOM)	`QUERY_HEADER,ROOM_ID,ROOM_TITLE,ROOM_TYPE,RO`
	`OM_MODE\r\n`
	`QUERY,17,Master Bedroom,LIGHT,0\r\n`
	`QUERY,14,Master Dress,BLIND,0\r\n`
	`QUERY,13,"One Room",LIGHT,0\r\n`
	`QUERY_COMPLETE,3`

Response description (ROOM)	*(see table below)*

Argument	Name	Example
0	ROOM_ID	17
1	ROOM_TITLE	Master Bedroom
2	ROOM_TYPE	LIGHT
3	ROOM_MODE *(Added in V3)*	4+OFF

ROOM_TYPE

LIGHT	Scene 1-4, Fade Up, Fade Down (Fade buttons should send stop on release)
BLIND	3 buttons open,stop,close. *(see table below)* If press/release events are unavailable use Alt.
SWITCH	On: Scene 1 Off: Scene 0 (Off)
CURTAIN	Same as Blind (Different room icon)
AUDIO	Deprecate RFU
BLIND_SMART	Positional blind with level percent used for blind position
CURTAIN_SMART	Same as BLIND SMART (Different room icon)
VENTILATION	Same as LIGHT (Different room icon)

Button	Press	Release	Alt
Open	FadeUp	Stop	Scene 2
Stop		Scene 3	
Close	Fade Down	Stop	Scene 4

ROOM_MODE

0	Default (None set)
S4OFF	4 Scenes, fade up/down, off
S7OFF	7 Scenes, fade up/down, off
S16OFF	16 Scenes, fade up/down, off
SNAMEDOFF	Named Scenes, fade up/down, off. Display only scenes outputted from query,scenes

Channel Query

Request Example	QUERY,CHANNEL\r\n QUERY,CHANNEL,17\r\n
Response example (CHANNEL)	QUERY_HEADER,ROOM_ID,ROOM_TITLE,ROOM_TYPE,ROOM_MODE,CHANNEL_ID,CHANNEL_TITLE,CHANNEL_TYPE,S1,S2,S3,S4,S5,S6,S7,S8,S9,S10,S11,S12,S13,S14,S15,S16\r\n QUERY,17,Master Bedroom,Lights,1,Ceiling,SLIDER,79,191,127,63,0,0,0,0,0,0,0,0,0,0,0\r\n QUERY,17,Master Bedroom,Lights,2,Bedsides,SLIDER,79,191,127,63,0,0,0,0,0,0,0,0,0,0,0\r\n QUERY_COMPLETE,2\r\n
Response description (CHANNEL)	 See table below

Argument	Name	Example
0	ROOM_ID	17
1	ROOM_TITLE	Master Bedroom
2	ROOM_TYPE *(Changed in V3)*	LIGHT
3	ROOM_MODE *(Added in V3)*	4+OFF
4	CHANNEL_ID	5
5	CHANNEL_TITLE	Pendant
6	CHANNEL_TYPE	BLIND
7..23	S1..S16	255

Channel Types:
SLIDER
BLIND
SWITCH
SCENES
FADEUPDOWN

S1..S16
Scene 1-16 for each channel (0-255)

Note	Query channel contains all data in QUERY,ROOM

CHANNEL_TYPE

SLIDER	Slider displaying 0 - 100% with data 0-255
BLIND	3 buttons open,stop,close. | Button	Press
Open	FadeUp
Stop	Scene 0 or 3
Close	Fade Down | The Rako iOS and Android app by default send scene 0 (Off) but have an option to send scene 3. Ideally stop on release for Open/Close but if not available can be missed.
SWITCH	On: Scene 1 Off: Scene 0 (Off)
SCENES	Off & Scenes 1-4
FADEUPDOWN	Fade Up, Fade Down. Stop on release (required).

Note: SCENES & FADEUPDOWN are rarely used.

Level Query

| **Response example (LEVEL)** | ```
QUERY,LEVEL,18\r\n
QUERY_HEADER,ROOM_ID,CHANNEL_ID,CURRENT_SCEN
E,CURRENT_LEVEL,TARGET_LEVEL,RFU\r\n
QUERY,18,0,-1,0,0,0\r\n
QUERY,18,2,-1,50,50,0\r\n
QUERY_COMPLETE,2\r\n
``` |
| **Response description (LEVEL)** | |

Arg	Name	Description
0	ROOM_ID	Room number
1	CHANNEL_ID	Channel number
2	CURRENT_SCENE	Current scene. Can be -1 if not set.
3	CURRENT_LEVEL	Current level
4	TARGET_LEVEL	Target Level
5	RFU	Refrain from use

Scene Query

Response example (SCENE)	`QUERY,SCENE,2\r\n`
	`QUERY_HEADER,ROOM_ID,SCENE_ID,SCENE_TITLE\r\n`
	`QUERY,2,1,Casual\r\n`
	`QUERY,2,2,Formal\r\n`
	`QUERY,2,3,Calming\r\n`
	`QUERY,2,4,Candles\r\n`
	`QUERY_COMPLETE,4r\n`

Response description (SCENE)	

Arg	Name	Description
0	ROOM_ID	Room number
1	SCENE_ID	Scene number
2	SCENE_TITLE	Scene title

RGB Query

Response example (RGB)	`QUERY,RGB,18\r\n`
	`ROOM_ID,ROOM_TITLE,CHANNEL_ID,CHANNEL_TITLE,`
	`RGB_INDEX,RGB_TITLE,RGB_COLOR\r\n`
	`QUERY,9,Kitchen,3,Plinth`
	`red,0,Plinth,RED\r\n`
	`QUERY,9,Kitchen,5,Plinth`
	`green,0,Plinth,GREEN\r\n`
	`QUERY,9,Kitchen,4,Plinth`
	`blue,0,Plinth,BLUE\r\n`
	`QUERY_COMPLETE,3\r\n`

Response description (RGB)	<table><tr><th>Arg</th><th>Name</th><th>Description</th></tr><tr><td>0</td><td>ROOM_ID</td><td>Room number</td></tr><tr><td>1</td><td>ROOM_TITLE</td><td>Room title</td></tr><tr><td>2</td><td>CHANNEL_ID</td><td>Channel number</td></tr><tr><td>3</td><td>CHANNEL_TITLE</td><td>Channel title</td></tr><tr><td>4</td><td>RGB_INDEX</td><td>Index of separate RGB areas in room</td></tr><tr><td>5</td><td>RGB_TITLE</td><td>A non unique title for the particular RGB area. (Can be blank)</td></tr><tr><td>6</td><td>RGB_COLOR</td><td>RED, GREEN, BLUE or WHITE</td></tr></table> Note: ● RGB areas are shown as colour wheels in the Rako app ● Items will contain RGB or RGBW ● Internally colour wheels are indexed by the room & channel value of RED. ● Send commands to hub using the SEND,RGB command (White channel is automatically calculated)

BASIC Feedback (Hub -> Client)

Feedback

Name	Feedback		
Feedback Example	`F,17,0,SCENE,1`		
Data			

Arg	Name	Example
0	Feedback	F
1	Room number	17
2	Channel number	0
3	Type	SCENE
4	Argument	1

Tracker

Name	Tracker (LEV)
Tracker Example	`T,17,8,LEV,127,79,299,0`
Data	

Arg	Name	Example
0	Tracker	T
1	Room number	17
2	Channel number	8
3	Type	LEV
4	Current Level	127
5	Target Level	79
6	Time to take (ms)	299
7	Temporary	0

Note:
- Temporary indicates it has come from a source such as fade buttons and is likely to not progress to the target level. If unsure, ignore events with temporary=1.

Name	Tracker (RGB)
Tracker Example	`T,17,8,RGB,127,79,299,0`
Data	

Arg	Name	Example
0	Tracker	T
1	Room number	17
2	Channel number	8
3	Type	RGB
4	Red	127
5	Green	79
6	Blue	299

Note:
- Red, Green, Blue values are 0-255
- The channel number represents the red channel in the RGB group

BASIC Notes

- The column names will only change with a new version specified with the 'SUB' command. It is not required to check the 'Column headers'.

[4 JSON](#)

Connection (JSON)

Name	**SUB (JSON)**		
Direction	Client->Hub		
Parameters			
	Arg	Name	Example
	0	Command	SUB
	1	Format	JSON
	2	JSON Structure	
Description	Subscriptions: **TRACKER**: Tracker events **FEEDBACK**: Feedback events		

Example	`SUB,JSON,{"version": 2, "client_name":`
	`"MY_TEST_CLIENT", "subscriptions":`
	`["TRACKER", "FEEDBACK"] }\r\n`		

Example Description	Subscribe as MY_TEST_CLIENT JSON format version 2 to Tracker and Feedback.		

STATUS (JSON)

Name	STATUS (JSON)
Direction	Client->Hub
Description	Subscriptions: **TRACKER**: Tracker events **FEEDBACK**: Feedback events

Example	`{       "name": "status",`
	`        "payload": {}`
	`}`

Response	<table><tr><td>Name</td><td>Example</td></tr><tr><td>productType</td><td>Hub</td></tr><tr><td>protocolVersion</td><td>2</td></tr><tr><td>hubId 36 character UUID</td><td>ebbe7961-7abb-3aed-9fef-0bb7871ef74d</td></tr><tr><td>mac</td><td>70:B3:D5:08:40:00</td></tr><tr><td>hubVersion</td><td>3.0.6</td></tr></table>
Example Response	`{"name":"status","payload":{"productType":"Hub","protocolVersion":2,"hubId":"ebbe7961-7abb-3aed-9fef-0bb7871ef74d","mac;":"70:B3:D5:08:43:27","hubVersion":"3.1.5"}}`

QUERY (JSON)

Name	QUERY (JSON)
Direction	Client->Hub
Parameters	<table><tr><td>Arg</td><td>Name</td><td>Example</td></tr><tr><td>0</td><td>queryType</td><td>ROOM</td></tr><tr><td>1</td><td>roomId</td><td>15</td></tr></table>
Description	Queries: **ROOM**: Query room names and types **SCENE:** Query scene names **CHANNEL**: Query room data and channel data **SCENECHANNEL:** Query SCENE and CHANNEL **LEVEL**: Query 'live' tracked level information **COLOR**: Query RGB/W and colour temperature items.

Example	<pre>{ "name": "query",
	        "payload": {
	                "queryType": "ROOM",
	                "roomId": 0
	        }
	}</pre>

Example Description	Query roomId 0 will return all rooms.

Response Example (ROOM)	<pre>{
	  "name": "query_ROOM",
	  "payload": [
	    {
	      "roomId": 9,
	      "title": "Test room",
	      "type": "LIGHT"
	    },
	    {
	      "roomId": 10,
	      "title": "Test room",
	      "type": "CURTAIN"
	    }
	  ]
	}</pre>

Response Example (CHANNEL)	<pre>{
	  "name": "query_CHANNEL",
	  "payload": [
	    {</pre>

```json
      "roomId": 9,
      "title": "Test room",
      "type": "LIGHT",
      "channel": [
        {
          "channelId": 1,
          "title": "Pendant",
          "type": "SLIDER",
          "sceneLevels": [0 ,255 ,191 ,127
,63 ,0 ,0 ,0 ,0 ,0 ,0 ,0 ,0 ,0 ,0 ,0 ,0]
        },
        {
          "channelId": 2,
          "title": "Bookshelf Red",
          "type": "SLIDER",
          "sceneLevels": [0 ,255 ,191 ,127
,63 ,0 ,0 ,0 ,0 ,0 ,0 ,0 ,0 ,0 ,0 ,0 ,0]
        },
        {
          "channelId": 3,
          "title": "Bookshelf Blue",
          "type": "SLIDER",
          "sceneLevels": [0 ,255 ,191 ,127
,63 ,0 ,0 ,0 ,0 ,0 ,0 ,0 ,0 ,0 ,0 ,0 ,0 ]
        },
        {
          "channelId": 4,
          "title": "Bookshelf Green",
          "type": "SLIDER",
          "sceneLevels": [0 ,255 ,191 ,127
,63 ,0 ,0 ,0 ,0 ,0 ,0 ,0 ,0 ,0 ,0 ,0 ,0 ]
        }
      ]
    },
    {
      "roomId": 10,
      "title": "Test room",
      "type": "CURTAIN",
      "channel": [
        {
          "channelId": 1,
          "title": "Window",
          "type": "BLIND",
          "sceneLevels": [0 ,255 ,191 ,127
,63 ,0 ,0 ,0 ,0 ,0 ,0 ,0 ,0 ,0 ,0 ,0 ,0 ]
        }
      ]
    }
  ]
}
```

| **Response Example (LEVEL)** | ```json
{
  "name": "query_LEVEL",
  "payload": [
    {
      "channel": [
        {
          "channelId": 0,
          "currentLevel": 50,
          "targetLevel": null
        },
        {
          "channelId": 1,
          "currentLevel": 50,
          "targetLevel": null
        }
      ],
      "roomId": 45,
      "currentScene": -1
    }
  ]
}
``` |
| **Response Example (SCENE)** | ```json
{
  "name": "query_SCENE",
  "payload": [
    {
      "roomId": 179,
      "title": "Garden",
      "type": "LIGHT",
      "mode": "S4OFF",
      "scene": [
        {
          "sceneId": 1,
          "title": "All On"
        },
        {
          "sceneId": 2,
          "title": "Wall Lights"
        },
        {
          "sceneId": 3,
          "title": "Garden"
        }
      ]
    }
  ]
}
``` |
| **Response Example (COLOR)** | ```json
{
  "name": "query_COLOR",
  "payload": [
``` |

```json
    {
      "roomId": 9,
      "title": "Test room",
      "type": "LED",
      "color": [
        {
          "sceneId": 0,
          "rgb": [0 ,0 ,0],
          "hex": "000000",
          "kelvin": null
        },
        {
          "sceneId": 1,
          "rgb": [255 ,255 ,255],
          "hex": "FFFFFF",
          "kelvin": null
        },
        {
          "sceneId": 2,
          "rgb": [191 ,191 ,191],
          "hex": "BFBFBF",
          "kelvin": null
        },
        {
          "sceneId": 3,
          "rgb": [127 ,127 ,127],
          "hex": "7F7F7F",
          "kelvin": null
        },
        {
          "sceneId": 4,
          "rgb": [63 ,63 ,63],
          "hex": "3F3F3F",
          "kelvin": null
        }
      ],
      "channel": [
        {
          "channelId": 2,
          "title": "Bookshelf Red",
          "rgbColor": "RED",
          "sceneLevels": [0 ,255 ,191 ,127
,63 ,0 ,0 ,0 ,0 ,0 ,0 ,0 ,0 ,0 ,0 ,0]
        },
        {
          "channelId": 4,
          "title": "Bookshelf Green",
          "rgbColor": "GREEN",
          "sceneLevels": [0 ,255 ,191 ,127
,63 ,0 ,0 ,0 ,0 ,0 ,0 ,0 ,0 ,0 ,0 ,0]
```

```
            },
            {
              "channelId": 3,
              "title": "Bookshelf Blue",
              "rgbColor": "BLUE",
              "sceneLevels": [0 ,255 ,191 ,127
,63 ,0 ,0 ,0 ,0 ,0 ,0 ,0 ,0 ,0 ,0 ,0]
            }
          ],
          "primaryChannel": 2,
          "rgbMode": "MULTICHANNEL",
          "rgbTitle": "Bookshelf"
      }
    ]
}
```

Query notes:

- Room & Channel types are defined in the BASIC section
- Query CHANNEL contains all information in Query,ROOM.
- Query,LEVEL will not return any rooms at power up, they will be tracked from first user interaction.
- QUERY,COLOR finds rgb/w capable rooms by channel names. The channels must be sequential containing 'red', 'green & 'blue' in their names.

SEND COMMAND (JSON)

Name	Send
Direction	Client->Hub
Payload	Payload:

Payload:

Element name	Example
room	Room number
channel	Channel number
description	Description
action	Action

Action:

Element name	Example
command	*required command
scene	***optional** scene number
level	***optional** level 0-255
down	***optional** fade direction

Example (Scene)	

```
{       "name": "send",
        "payload": {
                "room": 1,
                "channel": 2,
                "action": {
                        "command": "scene",
                        "scene": 3
                }
        }
}
```

Example (Level)	

```
{       "name": "send",
        "payload": {
                "room": 1,
                "channel": 2,
                "action": {
                        "command": "levelrate",
                        "level": 255
                }
        }
}
```

Example (Store)	```{ "name": "send", "payload": { "room": 1, "channel": 2, "action": { "command": "store" } } }```

Command list (JSON)

Command	ID	Argument	Description
ident	8		Flash the circuit
store	13		Store
stop	15		Stop fading
scene	49	scene	Scene 0-16 (0 = off)
levelrate	52	level	Level (0-255)
fade	50	down	Start fading up or down.

SEND COLOR (JSON)

Name	Send Color (RGB/W)
Direction	Client->Hub
Payload	Payload: | Element name	Example
room	Room number
channel	Channel number
red	Red (0-255)
green	Green (0-255)
blue	Blue (0-255 | Note: The channel number should be set to a single channel containing RGB. Channel 0 is also valid.
Example	```{ "name": "send-color", "payload": { "room": 16, "channel": 2, "red": 25, "green": 50, "blue": 255 } }```

Name	Send Color (Temperature)
Payload	Payload: | Element name	Example
room	Room number
channel	Channel number
temperature	Kelvin value (1700-10,000)	
Example	```{ "name": "send-colorTemp", "payload": { "room": 16, "channel": 2, "temperature": 2700```

| | ``` 
        }
}
``` |

Feedback

Feedback (JSON)

Name	**Feedback** (Use for mapping to button presses)
Direction	Hub->Client
Payload	

Element name	Example
room	Room number
channel	Channel number
description	Description
action	Action

Action structure

Element name	Example
command	*required command
scene	***optional** scene number
level	***optional** level 0-255
down	***optional** fade direction

```
{
  "name": "feedback",
  "payload": {
    "action": {
      "actUniqueId": -1,
      "defaultFadeRate": true,
      "decay": 0,
      "level": 90,
      "expFadeRate": false,
      "command": 52
    },
    "room": 85,
    "channel": 4,
    "description": "[Rm:85 Wine Cellar][Ch:4
Counter] Level: 35%"
  }
}
```

Tracker (JSON)

Name	**Tracker** (Use for tracking scenes and levels)
Direction	Hub->Client
Payload	

Element name	Description
`roomId`	Room number
`channelId`	Channel number
`currentLevel`	Current Level
`targetLevel`	Target Level
`timeToTake`	Time to take in milliseconds
`temporary`	Temporary indicates it has come from a source such as fade buttons and is likely to not progress to the target level. If unsure, ignore events with temporary=true.

Element name	Description
`roomId`	Room number
`channelId`	Channel number
`scene`	Scene number

Example (Level tracker)	

```
{
  "name": "tracker",
  "payload": {
    "roomId": 85,
    "channelId": 4,
    "currentLevel": 127,
    "targetLevel": 90,
    "timeToTake": 230,
    "temporary": false
  }
}
```

| **Example 2 (Scene tracker)** | ```json
{
  "name": "tracker",
  "payload": {
    "roomId": 85,
    "channelId": 0,
    "scene": 4
  }
}
``` |

5 UDP/IP

Products:

     RA/RTC/WA/WTC-Bridge
     APR/WRE-Bridge (V1.7.9 Limited support)

UDP Discover

All text is encoded with WINDOWS-1252

To find a BRIDGE or HUB on the network send a UDP broadcast packet where the data consists of a single literal 'D'. The bridge will reply to the source IP address with:

Request

Direction	Parameters		Description	Example
	Byte	Function		
Client to UDP Broadcast	0	`0x44`: 'D' for request	Discover Required	`0x44`
Client to UDP Broadcast	1	`0x58`: 'X' for request	Bypass Disable auto detect	`0x58`
Client to UDP Broadcast	2	`0x48`: 'H' for request	Append `\r\nHub\r\n2`	`0x48`

Bridges and HUBs are discovered by the same command so it is recommended to send "DXH" as the discovery command.

Response

Line	Description	Example
0	User configurable device name	`RakoHub`
1	Mac address	`70-B3-D5-08-41-9F`
2	Product type	`HUB`
3	Minimum api version	`2`

Example

Direction	Broadcast Reply From Bridge
Bridge	`RAKOBRIDGE\r\n00-04-A3-B1-21-F1\r\n`
HUB before version 3.0.6	`RakoHub\r\n70-B3-D5-08-40-00\r\nHUB\r\n`
HUB after version 3.0.6	`MyRakoHub\r\n70-B3-D5-08-41-9F\r\nHUB\r\n2\r\n`

HUB versions after 3.0.6 include the version. The TCP API requires 3.0.6 and later.

Note

The device name is a case insensitive alphanumeric string including space and hyphen with a maximum length of 16 characters.

6 Compatibility

Hub Version	Date	Description
Pre 3.0.6		API not supported.
3.0.6	18/12/2019	API V2 supported
3.0.9	27/04/2020	Issue present requiring extra arguments to send command (Use of version not recommended).
3.1.0	11/06/2020	API V3 supported. (V2 still supported).
3.1.5	17/08/2020	Minimum version for JSON support
3.1.6	21/08/2020	Support for RGB feedback
3.1.7	11/12/2020	Store command added to BASIC protocol. Added scene query to BASIC and JSON.

Note:
- All Hubs can upgrade to the latest version.
- RA/WA/RTC/WTC/WRA Bridge will never support this protocol. Use 'Accessing The Rako Bridge'.

7 Changelog

Version	Date	Description
0.1.5	21/01/202	API version V2
0.1.6	24/04/2020	BASIC API V3 ● Added changelog ● PLAIN renamed to BASIC (Only in document) ● Documentation for JSON API ● Added description for RoomType and ChannelType ● Added room mode to Room query ● V3 QUERY,CHANNEL Room Type was changed to be the same as QUERY,ROOM ● Added QUERY,RGB ● Added SEND,{room},{channel},RGB,{R},{G},{B}
0.1.7	20/06/2020	Document formatting
0.1.8	10/08/2020	Added JSON query information
0.1.9	13/08/2020	Query example correct for Room and Channel
0.2.0	25/08/2020	Update UDP Discovery descriptions
0.2.1	02/12/2020	Added Store command Added scene query to BASIC and JSON
0.2.2	18/02/2021	Basic tracker returns 'LEV' not 'LEVEL' Basic tracker 'RGB' type
0.2.3	23/04/2021	Updated Introduction Added missing JSON query scene