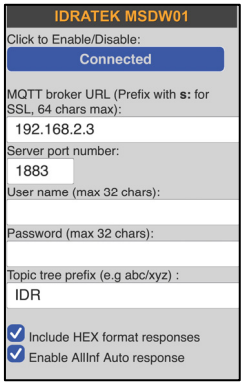
MQTT Explorer and MSD-W01 Examples



This is a simple example illustrating the use of the open source MQTT Explorer client (https://mqtt-explorer.com/) to interact with an MSD-W01 module for MQTT experimentation. The physical set up in this case is that this MSD module is actually part of a wider IDRATEK network, with Cortex interacting at the same time with this same module as well as others directly using the Cortex IOTA-WiFi option. A local MQTT broker (Eclipse Mosquito - https://mosquitto.org/) is running on a small computer platform on the same LAN as Cortex and the MSD module. The MSD module is set up to connect to the mosquitto broker and MQTT Explorer is also connected to the same broker. In the module's MQTT settings, both HEX format and 'AllInf' Auto response options are enabled. The latter generates a JSON formatted message containing details about the module and its various signal states whenever there is either deliberate module interrogation or as a result of any internally enabled trigger sources – for example could be due to a sufficient temperature change, a motion detection state change, etc. Note that a variety of such triggers are usually enabled by Cortex so that any signal changes are captured at the appropriate time rather than relying on frequent polling. The JSON formatted MQTT payload format is well suited for parsing by the MQTT Explorer application which also very conveniently allows the changes in signals to be easily displayed in graphical form. Below we can see a selection of available signals from this single module cast into such graphs, each with a 15 minute time scale.

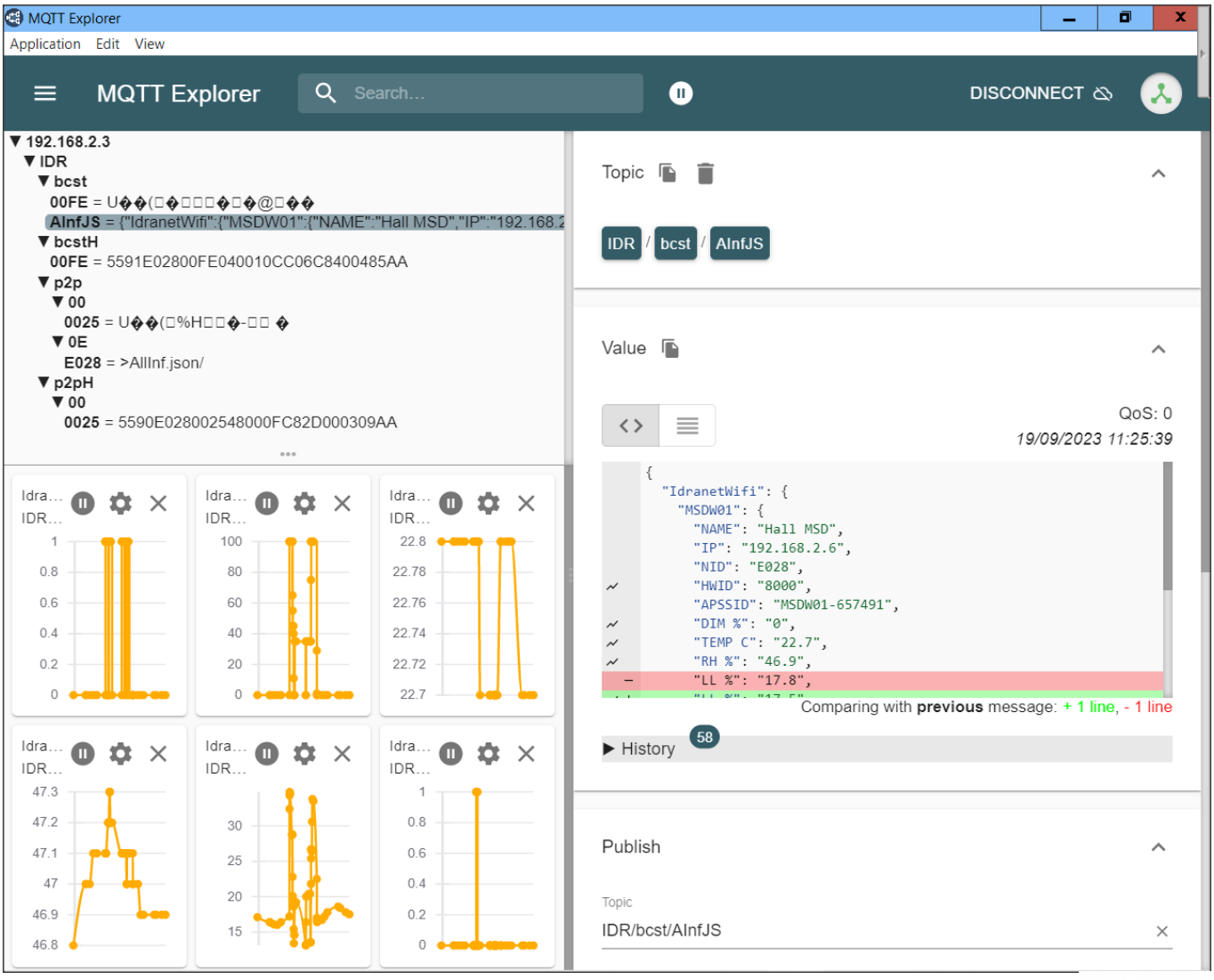
Topics and Payloads

If you are unfamiliar with how MQTT is utilised by IDRATEK modules then it may help to read the MQTT related sections of the MSD-W01 guide document first.

The screen shot below shows a typical set of Topics and Payloads associated with the data. We can see in this that data was transmitted to a variety of Topic paths depending on its nature and the reporting options enabled. For example payloads sent to IDR/bcst/00FE are raw data byte packets which are being broadcast to any devices which accept a ZIDTID address of 00FE (Cortex). These types of packets are generated by any enabled module 'Auto Response' reflexes – meaning the module automatically sends data when triggered by a particular signal change. In this case we also see the ASCII Hex form of the same data being sent to IDR/bcstH/00FE. The data identifier within this packet (CC) indicates a humidity measurement with raw value of 06C8.

Also evident in this set of messages is an outgoing MQTT message, being sent manually from MQTT Explorer to the MSD module. In this case the Topic is IDR/p2p/0E/E028 (since the MSD module NID is E028 in this case) and the payload is >AllInf.json/ (meaning a manual request for the AllInf JSON response). This response is always sent to the /bcst/AInfJS sub topic.

Another interesting observation is a message to IDR/p2pH/00/0025. The topic implies the target has a NID of 0025. This is in fact the notional ID of the Cortex direct WiFi interface for this system (think of it as similar to a PCU module except the latter handles communications between Cortex and the wired network). In this case it would appear that Cortex has polled the MSD for light level data since the Payload has been sent using p2p to the Cortex NID rather than being a broadcast. The data identifier is C8 (meaning 8 bit Light level return) and the value is 2D

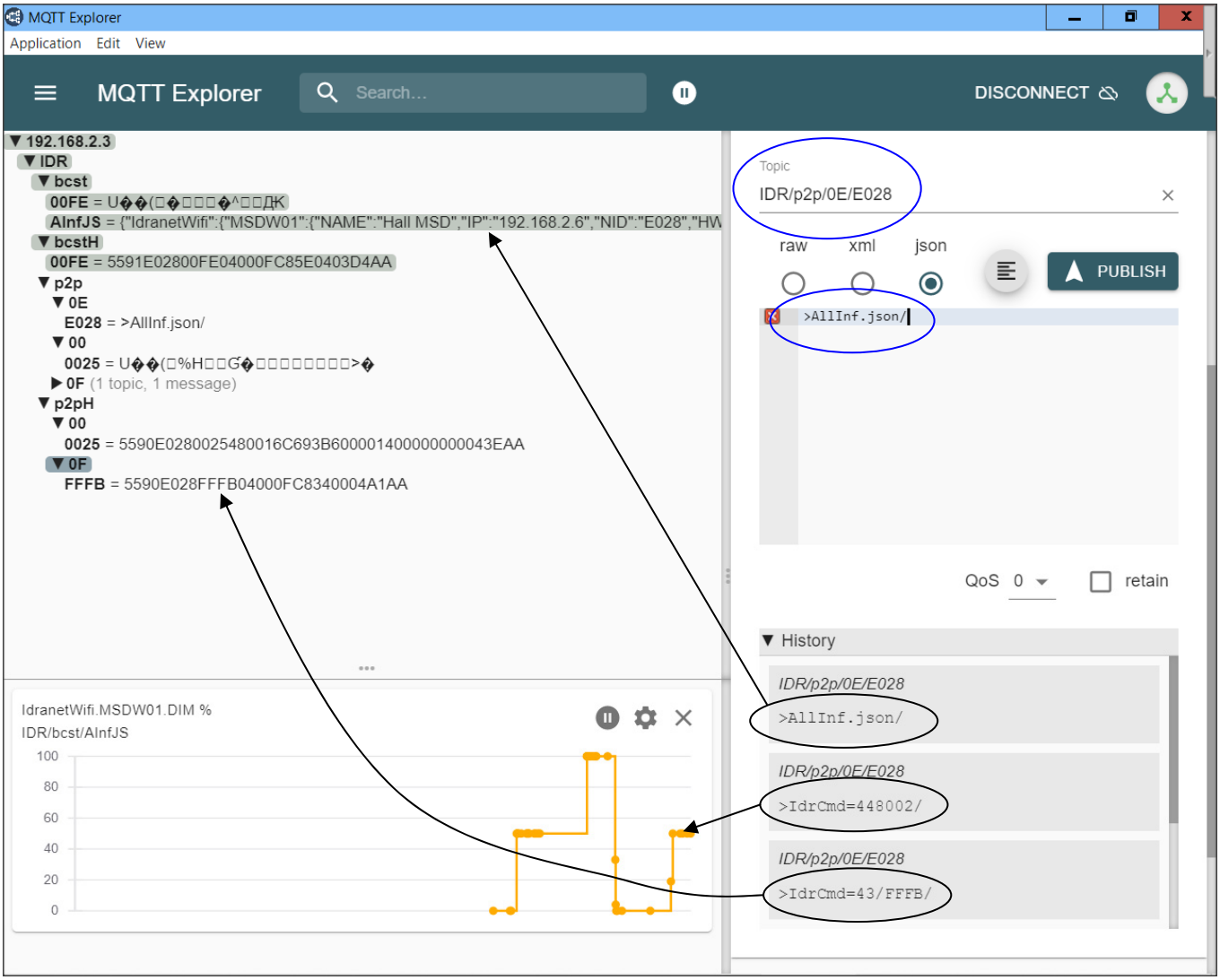


Sending commands from MQTT Explorer

Below is a screenshot illustrating the manual sending of commands to a module from MQTT Explorer. Here we have chosen to send the commands to the MSD module whose NID is **E028**, using **p2p** addressing (meaning targeting that module specifically). The Topic prefix in this experimental system is just **IDR** so, according to the IDRATEK rules, the Topic path is constructed as **IDR/p2p/0E/E028**

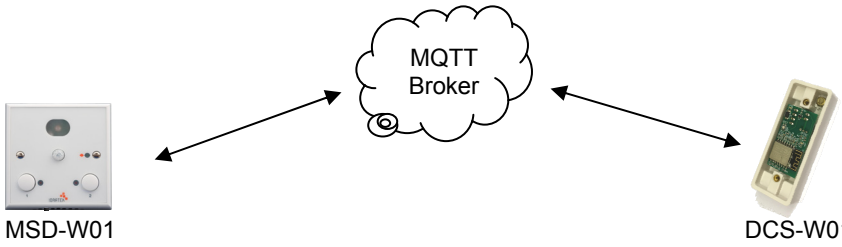
In the History listing we can see some of the commands that were sent to the module and their effect can be seen in the MQTT message listings and/or in the graphical rendition, in this case for brevity, just of the Dimmer output value.

We can also see within this list, a command requesting light level data from the module and using the 'sender's NID' appended command syntax, such that the response from the module to this request targets a notional IDRANet NID of **FFFB**



Example of eavesdropping on Reflex MQTT activity

Below we see an example where an MSD module is using purely IDRATEK Reflex programming to operate the output on a remotely located DCS board module. The Reflex program in the MSD module is configured to send two different command packets to the DCS module depending on the direction of change of the motion sensor state – such that when the motion sensor goes ON the DCS output is commanded to ON and vice versa (thus the remote DCS output tracks the MSD sensor state). The communications are directly between the two modules via IOTA MQTT with no other IDRATEK hardware nor Cortex involved. On a local area network it would be possible to achieve the same functionality via the alternative IOTA UDP protocol without needing an MQTT broker. In fact in principle for such a limited application you would not even need a separate local router since one of the two modules can act as the router via its Access Point mode.



Here we are using MQTT Explorer to analyse the communications that are taking place. In the history listing we can see the IDRANet protocol command packets being sent from the MSD module (NID E024) in response to PIR state changes - targeting the DCS module (NID E004), with either a command (3E0102) to turn ON its output or (3E0202) to turn it OFF