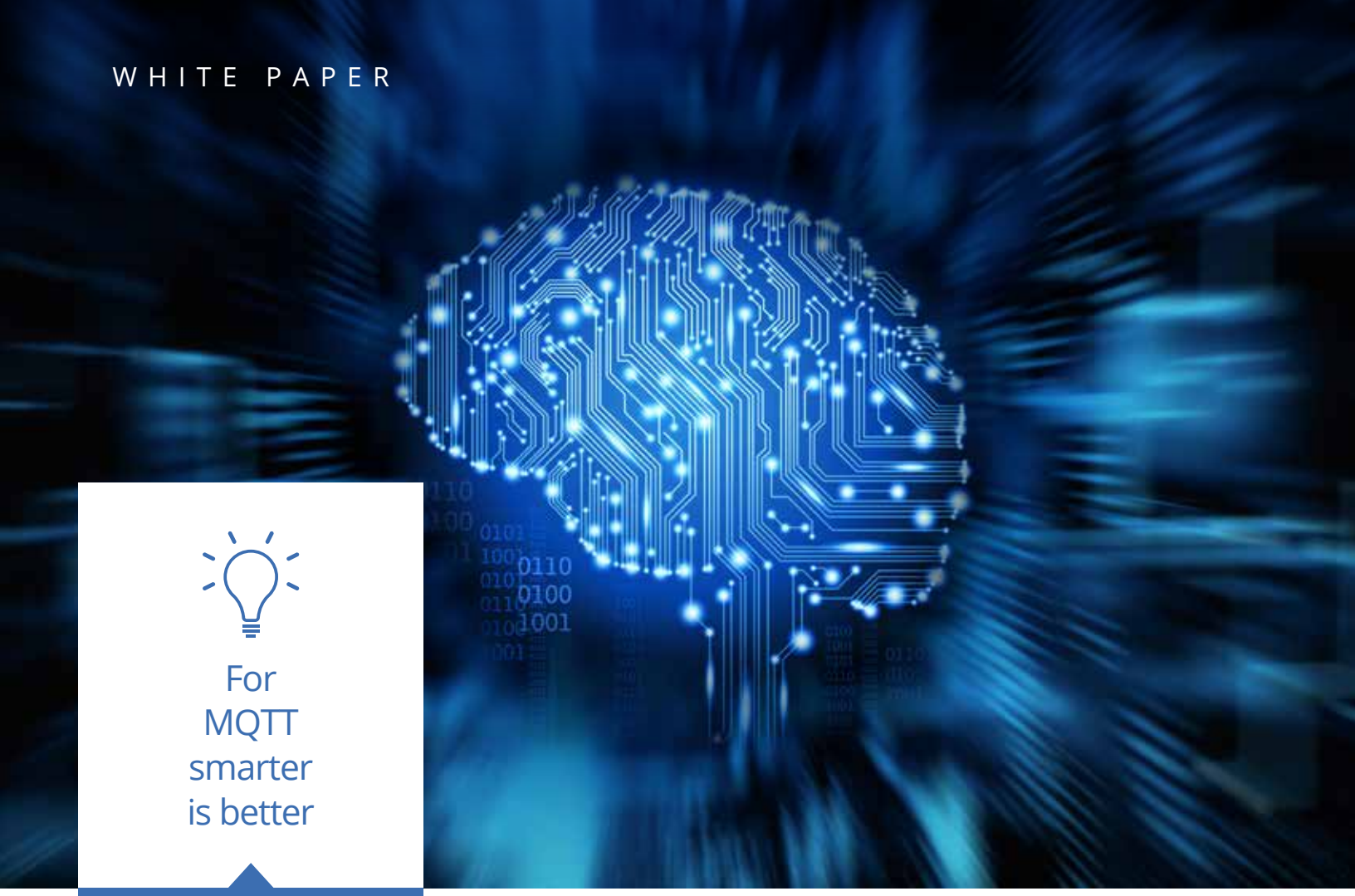
WHITE PAPER



For MQTT smarter is better

Cogent DataHub™

MQTT is the protocol of choice for many industrial communication tasks, particularly in the oil and gas sector. It was developed to be efficient, quick, and secure, and it delivers on those promises. It does a good job at what it was designed for—connecting field devices to a central SCADA system and passing data between them.

With the advent of Industrial IoT (IIoT), proponents of MQTT stepped up to offer it as a way to connect production data to the cloud. And with increasing interest in connecting OT (operations technology) to IT, MQTT has been called upon to connect not only sensors and actuators in the field but also edge devices, SCADA systems, IoT gateways and more. These get linked to various tools used by corporate IT departments including historians, data lakes, AI engines, and other analytical instruments.

Bigger challenges

This broader range of application spaces challenges the MQTT protocol that was intentionally kept simple to ensure speed and flexibility. Instead of each connection carrying data from a single device, MQTT is being called on to send collections of data values. Where once all devices may have been identical, now a variety of devices must communicate with each other using different data formats.

The simple, direct security model of device-to-client is not sufficient anymore when networks need to be isolated using DMZs, requiring multiple-hop connections. A new specification, Sparkplug, was introduced to meet some of these challenges, and yet there are ways that it too, can be enhanced.

Get smarter

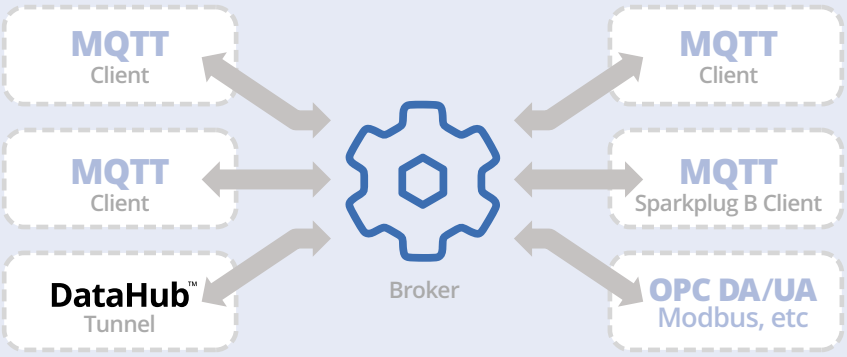
These challenges demand that MQTT get smarter. By design, MQTT is a transport protocol, like a postal service carrying letters. The service doesn't know or care what's in the letters and takes no interest in the personal lives of those who send or receive them. It just carries and delivers letters. Likewise, an MQTT broker has no knowledge of the content of its messages, nor the status of sender and receiver.

Now suppose we make the MQTT broker smart. What if we give it the ability to read and understand the messages it carries? Like the curious postal clerk that reads postcards while delivering them, the MQTT broker could now parse them and handle messages more intelligently. And what if the broker could communicate with the senders and receivers themselves? It could then inform them of network status or which clients may have disconnected. It could also talk to different MQTT formats, like JSON, Sparkplug or JSON schema and expose the needed format to the connecting client,

The DataHub™ Smart MQTT Broker™ is this kind of smart MQTT broker. It is invaluable for the growing demands being put upon MQTT. Let's look in more detail at what's needed, and how making MQTT smarter can make it better.

Data collection

For many IoT and OT-to-IT applications, the simple device-to-broker MQTT connection is not sufficient. On large-scale systems with hundreds or thousands of connected devices, the data streams may need to be consolidated into a few or even one MQTT connection. This is particularly true for cloud services that accept only one client connection, or that charge on a per-connection basis.



With this aggregation of data streams, different devices and data sources may need to be integrated. Although all may use MQTT, there is a good chance that they use different message types. And in many scenarios, MQTT is being integrated with other industrial protocols, such as OPC UA.

The DataHub Smart MQTT Broker provides native connectivity and data conversion from OPC UA, allowing it to collect and aggregate incoming data in this way. By parsing all incoming messages, it can translate between various MQTT message types and provide a single outbound MQTT message type. And it can read data in other common protocols like OPC DA and convert that data into the same MQTT message type as well.

Data consistency

In a real-time industrial system, data consistency is critical. An operator monitoring an HMI or SCADA system needs to know exactly what's happening on the physical device. Data that's stale or out of correct time sequence can lead to incorrect decisions. Also, any disconnects or network irregularities must be known. The DataHub Smart MQTT Broker leverages its ability to parse messages, along with smart message queueing, to ensure data consistency.

Smart message queueing is needed in real-time systems to handle message overload. This happens when a data producer, like a sensor or other device, sends data faster than a consumer can receive it. A chronic overload requires the broker to drop messages.

The DataHub Smart MQTT Broker implements an intelligent message queue that examines the message content and ensures that the latest value of every data item is delivered, even when earlier values are dropped.

This keeps data at the consumer consistent with the physical reality of the producer.

Latest value – Having the latest value of the data is critical in an industrial system. Suppose, for example, in a burst of activity a pump is switched on and off many times, with the final position being "OFF". If that final MQTT message gets dropped by the broker, the HMI or SCADA system will show the pump as "ON." This kind of inconsistent data can lead to costly errors and system malfunctions. A regular MQTT broker without smart message queueing may drop that final, latest value, whereas the DataHub Smart MQTT Broker with smart message queueing ensures that it gets delivered.

Time order is preserved in a single MQTT message topic, but not necessarily among multiple topics. Events coming from different devices that occur in the order A then B then C could be delivered to an application as C then B then A, or any other ordering, which is an error in many industrial control use cases. The DataHub Smart MQTT Broker preserves time order as it converts messages to other protocols for transmission to control systems or retransmission across a network.

Connection status – Regular MQTT brokers do not have a way to indicate that a data source is disconnected. The consuming application cannot tell the difference between an old value from a sensor that has failed, or a current value that has simply not changed recently. The "last will" mechanism in MQTT designed to deal with this requires unreasonable levels of coupling between the producers and consumers of data, resulting in duplicate configuration and increased integration and maintenance costs.

The DataHub Smart MQTT Broker monitors the condition of the data producers and the network, and assigns a quality code to each message, updating it with each value change. This information can be included in the outgoing MQTT message. This way data consumers have some way to tell why a value is not changing.

Data security

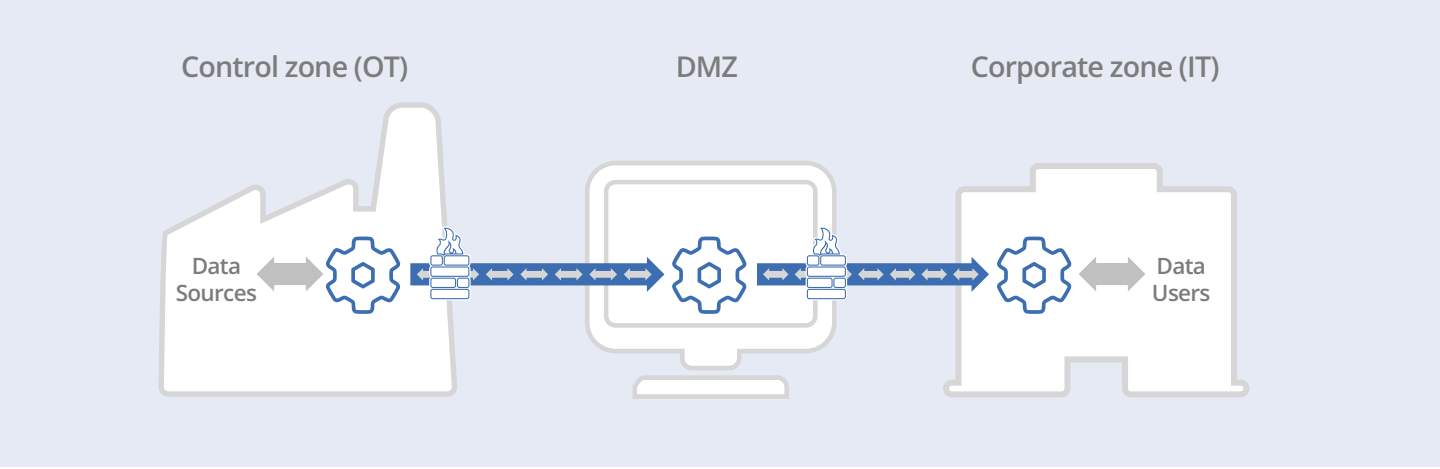
Industry security experts and government agencies recommend isolating networks for connecting OT and IT systems. The preferred approach is by using a DMZ. NIST document SP-800-82 sums up it up like this: "The most secure, manageable, and scalable control network and corporate network segregation architectures are typically based on a system with at least three zones, incorporating one or more DMZs."

These three zones are the control zone (OT), the corporate zone (IT), and the DMZ in the middle. Using a DMZ ensures that there is no direct link between corporate networks and control networks, and that only known and authenticated actors can enter the system at all. The SP-800-82 document describes the value and use of firewalls to separate these zones, and to ensure that only the correct data passes from one to the other.

Multi-hop daisy chain

Implementing data flow through a DMZ is problematic for MQTT, as this kind of connection typically requires two or more servers, linked together one after the other in a daisy chain. The QoS guarantees in MQTT cannot propagate through the chain, making data at the ends of the chain unreliable.

One reliable solution is to convert the MQTT message into DataHub Transfer Protocol (DHTP) format that can be passed over the network from server to server until it reaches its destination. The device producing the



MQTT data is connected to an instance of the DataHub Smart MQTT Broker. That DataHub instance does the data conversions and passes the data, along with its quality information, via DHTP to a second instance of the DataHub Smart MQTT Broker running on the DMZ. The data is then passed to a third DataHub instance running in the IT space, which converts the data back into MQTT.

To ensure secure data transfer, DHTP offers SSL encryption with support for the most recent versions (TLS 1.2 and TLS 1.3), along with using and enforcing server certificates. And to protect the OT and IT systems from unwanted intruders, the DataHub Smart MQTT Broker can send data outbound from a firewall without opening any inbound ports. It is critical that this valuable security feature of MQTT be retained.

MQTT Sparkplug enhancements

The Sparkplug specification for MQTT was introduced to resolve interoperability issues between vendors by defining how data is sent and received. Sparkplug classifies MQTT clients as either edge of network (EoN) devices that produce data, or as applications that consume data. Each Sparkplug device produces messages of various kinds, like a BIRTH message to show it has come online, DATA messages for sending data, and a DEATH message when it goes offline. Any Sparkplug application that is online receives these messages and is thus kept informed of which data is coming from which device.

All of the smart broker capabilities discussed so far apply to an MQTT Sparkplug-based system. Additionally, the DataHub Smart MQTT Broker provides other features to further enhance Sparkplug connectivity.

Synchronizing all Sparkplug 2.2 applications – Because it is aware of all connections, the DataHub Smart MQTT Broker can synthesize a BIRTH message for each connected device whenever a new Sparkplug 2.2 application comes online. This allows that application to receive DATA messages from all currently connected devices, eliminating issues related to start-up order.

Responding to errors – In addition to its ability to identify out-of-order or lost MQTT messages, the DataHub Smart MQTT Broker can automatically disconnect a Sparkplug device when these kinds of errors occur, and allow it to reconnect. This would cause the device to re-send its BIRTH (startup) message, which will resynchronize all receiving applications, thus maintaining a single version of the truth.

Resolving failed writes to devices – Another useful feature is to check all write requests from applications to devices, to ensure the specified data value was written on the device. If the DataHub Smart MQTT Broker detects the value on the device did not change, it can force the device to disconnect, causing it to retransmit its BIRTH message. This will resynchronize all applications listening to that device, and is another way to maintain a single version of the truth.

Adding data quality information – For systems that need to convert Sparkplug data to other protocols, the DataHub Smart MQTT Broker adds quality information. For example, when converting Sparkplug data to OPC, it adds OPC data quality. BIRTH or DATA messages are assigned the OPC data quality of Good, while DEATH (shutdown) messages take a Not Connected quality.

Getting better

As valuable as MQTT is for device-to-server data communication, it can get even better—to take on the challenges of OT/IT, Industrie 4.0, and the Industrial IoT. The DataHub Smart MQTT Broker collects data from multiple incoming message types, and even other protocols. It ensures data consistency over the entire path of the message from data producer to data consumer, where the consumer always has the latest value and with an indicator of data quality. The DataHub Smart MQTT Broker can also securely connect MQTT data producers and consumers across DMZs and other multi-hop network configurations. These advantages and more are on offer for MQTT Sparkplug implementations as well. For today's requirements and those that lie ahead, as MQTT gets smarter it gets better.

About Skkynet

Skkynet is a global leader in real-time software and services that allow companies to securely acquire, monitor, control, visualize, network and consolidate live process data in-plant or in the cloud. DataHub™, DataHub™ for Azure, and Embedded Toolkit (ETK) software enable secure, real-time data connectivity for industrial automation, Industrial IoT, and Industrie 4.0. Visit **skkynet.com** for more about the company and **cogentdatahub.com** for more about DataHub.