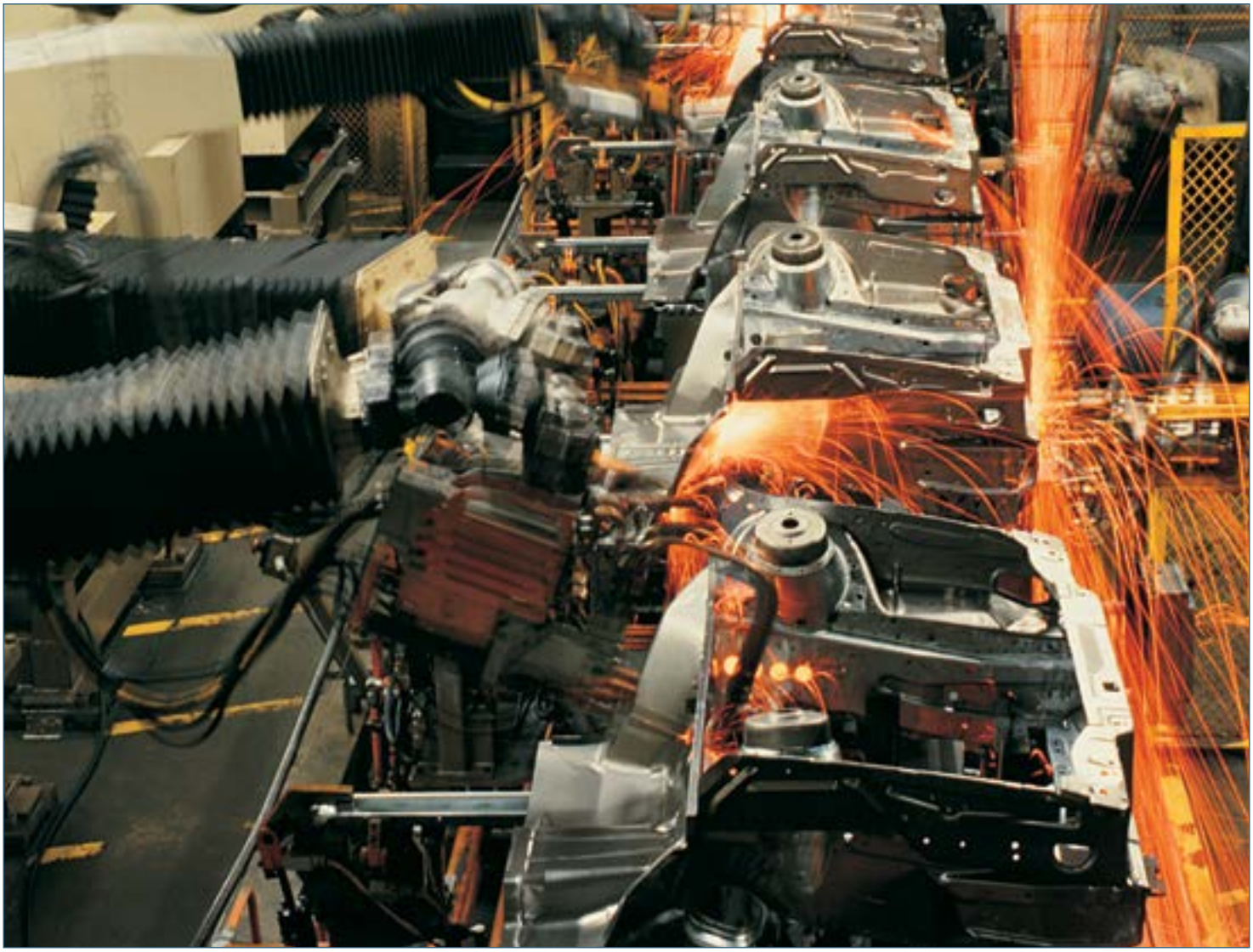
MQTT used in production - a use case



Whitepaper A quick look into an implementation of MQTT to realize Industry 4.0

www.hms-networks.com

Table of contents

1. Introduction

How can you overview your networked assets and collect diagnostics from them to be able to perform utilization analytics and for predicted maintenance purposes? And how can you use the data to track possible quality issues in the products?

This white paper is a fictual example of how the MQTT protocol can be used for pushing non-time critical data from the factory floor to the data fog in an enterprise network. The fictual company used in the case is an industrial manufacturer with production sites spread across the world.

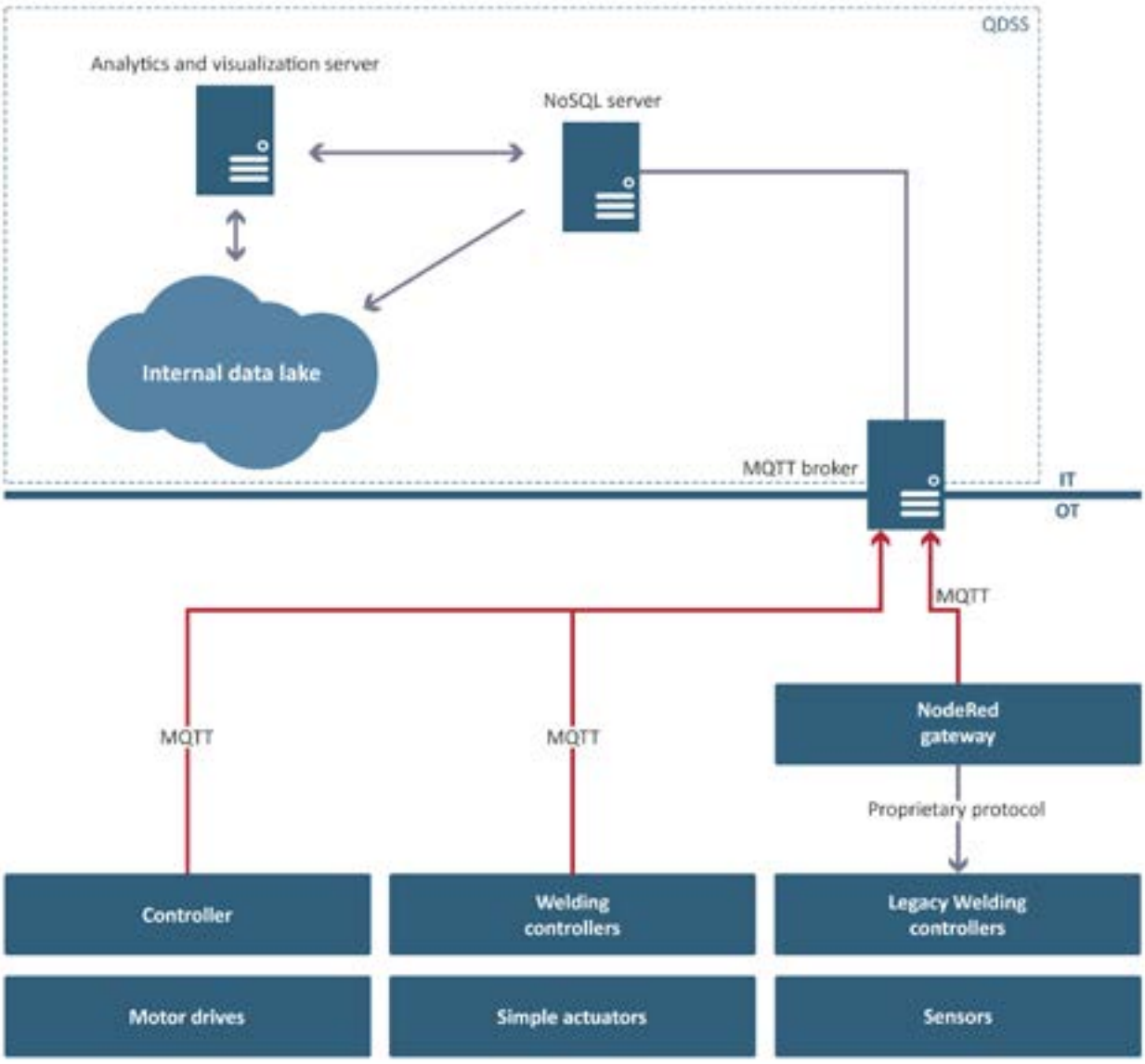
1.1 The goals

- Analyze factory utilization
- Perform predictive maintenance
- Track possible quality issues

The manufacturer in this example is producing big industry machines. One important part of the production is the welding of the machines. The company is collecting approximately 100 quality associated statistics for each welding spot on one of its products. And every product contains hundreds of welding spots. The manufacturer needs to store this quality associated data to be able to follow up on reclaims in the future, or to track possible quality issues on a product by product basis. Historically this type of quality associated data has been provided via proprietary solutions by each individual welding machine vendor.

But quality tracking is not all the manufacturer wants to get out of its factory data. Another goal is to get an overview of the networked assets to be able to perform utilization analytics and for predicted maintenance purposes. The data is also supposed to be used to oversee the whole production process.

The overall goal for this manufacturer is to combine production analytics, diagnostics and quality associated data (we will call it secondary data throughout the rest of the document) via one standardized production system.



2. A quick look at the system

The manufacturer has built an IT infrastructure that they call QDSS (Quality Data Storage System) that is based mostly on components from IBM for storing and perform analytics on the data. MQTT has been chosen as the standard transport protocol for the secondary data on the OT side of the system. It is not used for transportation between different systems on the IT side. The MQTT broker receives all data published by the OT devices and is customized to connect to a NoSQL server system using a non-MQTT interface.

When, for instance, a welding controller has welded a spot on a product it will produce an MQTT publication containing information about voltages, currents, temperatures etc. to an internal MQTT broker. The MQTT broker then forwards the data to QDSS using a customized interface to the NoSQL server.

Component examples for the system:

NoSQL server – IBM Cloudant internal

Analytics and visualization server – Apache Spark

MQTT broker* – Mosquito

*MQTT broker is using Client SDK on IT side to connect to Cloudant

The manufacturer anticipates that one production site alone can produce up to 2TB of data per day when all components in the factory are connected to the QDSS. To be able to handle the data volume QDSS has a short term and long term storage principle implemented. The analytics is done on JSON formatted data and QDSS will store the raw JSON data from each MQTT production in the NoSQL server for a shorter period (range of weeks). After that period the quality associated data is moved to an internal data lake for long term storage. Data that is not important for long term is deleted.

2.1 Aiming for full implementation

The ambition is that all Ethernet connected machines and devices on the production floor shall support MQTT natively. The support for MQTT is a requirement when acquiring new equipment to their factories. Currently the manufacturer is connecting all welding controllers to QDSS and the production technicians will be working on connecting more device types in the future. They will start with the most complex machines and will then work their way down to the smallest sensors and actuators.

One of the welding controller vendors has implemented support for MQTT natively, using an Anybus CompactCom40 design. The MQTT protocol is implemented using the CompactCom IIoT module with MQTT support.

Another welding controller vendor is currently not supporting MQTT natively. Their controllers store all secondary data in an SQL database. The manufacturer has built a simple gateway application on NodeRed that fetches data from the SQL database and produces appropriate MQTT publications to QDSS. This is a simple way to retrofit legacy devices into the system. The manufacturer sees this as an interim solution and will be working actively to get the vendors to implement MQTT natively.

3. How the MQTT protocol is used

The manufacturer can set a number of rules for the devices that shall be able to connect to the QDSS system via MQTT. In this example the manufacturer has decided that all devices shall implement the following manufacturer specific JSON messages:

- ONLINE_NOTICE – An Online notification that contains a time stamp for the online event.
- OFFLINE_NOTICE – An Offline notification using the Last will function in the MQTT protocol.
- ASSET_INFO – This message must be sent during start-up and contains identification and location data of the asset.

The Online and Offline notification messages are used to track each individual welding controller's online time. The Asset information message is used to describe the welding controller's location, the cell's name and identification data of the welding controller such as model number, serial number etc.

Except for the JSON requests above each device is free to define its own JSON messages based on the application type. The company has the ambition to try to group as much data into each MQTT message as possible. Instead of publishing data on data change they try to publish data based on an event.

For example, a welding controller can publish a JSON message for each weld spot. The message will contain diagnostic information about currents, voltages, temperatures etc. at that specific time.

```
Topic:
COMPANY-X/JSON/0/1/WELDING-STATION-12/WELD/1
JSON Payload:
"weldspot" = {          Timestamp="20181001 171035"
                Current="10",
                Voltage="230",
                MainPowerSwitch="0",
                MainPowerSwitch_str="OFF",
                …. }
```

Above is an example of how a JSON message describing a weld spot could look like.
It is very important for the manufacturer that as much meta data as possible is available in the JSON

message as they don't have any way to parse unstructured data in their IT system. Each value in the JSON message shall more or less be possible to understand without the need of translation or having to read the user manual to understand the purpose of a value. For enumeration values like the MainPowerSwitch example above both a numerical and a string representation is included in the JSON message. The numerical value is needed when trending a value in a chart diagram and the string representation is needed for human understanding.

The company decided to only use upper case topic strings. This is to avoid any problems with the case sensitive topic strings on MQTT. The format of the topic is strictly outlined and looks something like this: ROOT/FORMAT/COMPRESSED/PLANT-ID/DEVICE-ID/MSG-TYPE/MSG-VERSION. One important thing to do is to add message format version in the end of the topic string. By this the message payload format can be changed and evolved over time without having any format changes for a unique topic.

3.1 Two way communication raises the bar on security

QDSS is only built to transfer data from the edge devices to the IT system. It has no functionality to transfer data in the other direction. The IT system in this example is not built in a way to make this possible. However there are use cases for configuring devices on the factory floor via MQTT. By adding a possibility to configure a device via MQTT a security concern is introduced where the IT system and the entire connection to the end device must be secured.

3.2 Data size matters

The size of the MQTT publication for each welding spot is approximately 30kb. As long as the JSON payload is smaller than 100kb the manufacturer in this example intends to continue to use JSON. They have identified that JSON payloads larger than 100kb will start to cause problems both on the network bandwidth but also on QDSS. If any application will have payloads larger than 100kb a different encoding format will be evaluated, examples of this could be gzipped JSON, XML and binary protocols like protocol buffers for example.

With the large production sites, the MQTT broker will from time to time be very busy and there is a risk of overload. The company intends to solve this by adding a load balancer on the network that redirects MQTT messages to different brokers on the IT side, with no knowledge of the MQTT clients. As a simpler solution the manufacturer wants every MQTT client to be configurable with up to three different MQTT brokers. If the client loses connection with one of the brokers, it should automatically try to connect to a second one.

3.3 Dont retain old data

In the implementation of MQTT into QDSS all MQTT messages are using QoS level 0 with clean session bit always set to 1. Retain messages are not used at all. The reason for this is that they don’t want to have old data cached anywhere in the system. In this installation and environment, the TCP protocol is trusted to always deliver the MQTT frames to the broker.

4. Taking a stepwise approach to security

The IT system increases the security of the production plant but also introduces new security requirements on the system. One of the advantages is that diagnostic data is available for fault investigation directly in QDSS. This limits the need for service technicians to connect directly into the factory network.

By adding a connection between the factory network and the enterprise network a security risk is added. To mitigate this risk the long-term goal is to only allow secure protocols on the factory floor where data integrity and device authentication can be verified. This is not possible for most of the industrial networks today but for example EtherNet/IP and Modbus TCP are currently evolving their protocols by using TLS to accomplish data integrity.

The first step for the manufacturer is to have all MQTT connections to utilize TLS. Today a couple of vendors has this implemented but far from all. When TLS is not utilized there is no meaning to use the built-in user authentication mechanism in MQTT as it will be transferred in plain text over the network.