

WEBINAR

How MQTT 5 Makes Difficult IoT Use Cases Possible

HOSTED BY



HIVEMQ



Florian Raschbichler
Head of Support



Speaker



Florian Raschbichler
Head of Support at HiveMQ

 @fraschbi

 <https://www.linkedin.com/in/fraschbi/>

Florian serves as the head of the HiveMQ support team with years of first-hand experience overcoming challenges in achieving reliable, scalable, and secure IoT messaging for enterprise customers.



MQTT 5 – Quick Intro

The most feature-rich update of the MQTT protocol ever!

- Enhancement for scalability and large scale systems
- Better error handling for more robust systems
- More scalability for cloud-native computing
- Formalize common patterns including capability discovery and request response
- Performance improvements and support for small clients



Popular Use Cases



- Connected cars
- Home automation
- Manufacturing systems
- Transportation
- Logistics
- Enterprise chat applications
- Mobile apps



MQTT 5 – Rich Feature Set



- Topic Aliases
- Shared Subscriptions
- Request-Response Pattern
- User Properties
- Session and Message Expiry
- Payload Type Descriptors
- Enhanced Authentication



Deep Dive into Primary Feature Set of MQTT 5

Explained with Use Case Examples



Topic Aliases

Helps Save Bandwidth and Memory



Negotiated individually between client and broker



Certain topics get exchanged with integer alias instead of a string value

Benefits of Topic Aliases

Ensures low latency and limited data consumption for high frequency messaging

Removes bandwidth usage from network as the provider

Save data as the consumer



Topic Aliases

Helps Save Bandwidth and Memory

MQTT Clients



Publish

MQTT Broker

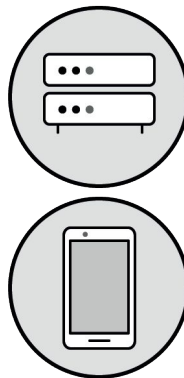


Subscribe

Subscribe


Publish

MQTT Clients



MQTT-Packet:

PUBLISH



contains:	Example
packetID	4314
topicName	" "
topicAlias	1
qos	1
payload	756



V2X Use Case

- Vehicle to Anything (V2X)
- High frequency messaging (10hz or more)
- Tens or hundreds of thousands of devices (cars, phones, sensors, traffic lights ..)
- Deep topic structure for services, layers and geo location
- Example calculation
 - Topic as String:
Europe/Sweden/Stockholm/North-East/Tile121/Priority/Pedastrian-Alert
= 68 bytes
 - Topic as Integer = 4 bytes
 - $64 \text{ bytes} * 10.000 \text{ cars} * 10\text{hz} \sim 6 \text{ MB/s}$
=> ~ **15 TeraByte per month for 1 sensor**
- Useful for OEMs, ISPs and service provides (cities)



Improved Error Reporting

Multiple new return codes

Introduction of negative acknowledgement

Optional, arbitrary ReasonString



Shared Subscriptions

Multiple clients can share the load on a topic

Client load balancing mechanism

Special syntax



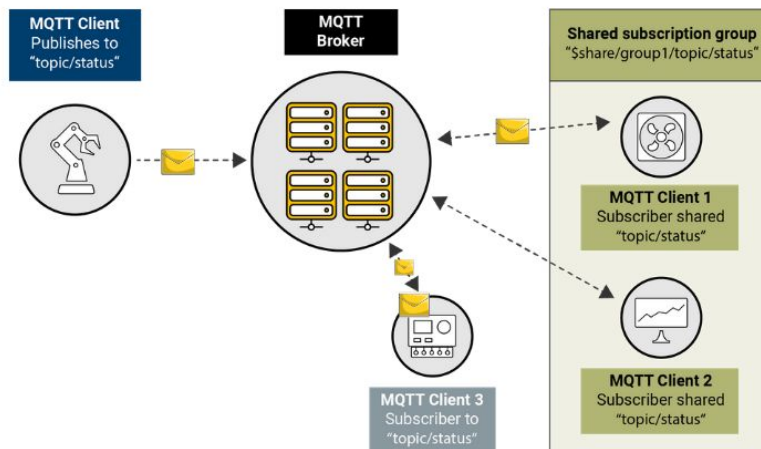
Shared Subscriptions

`$share/GROUPID/TOPIC`

Shared subscription
identifier

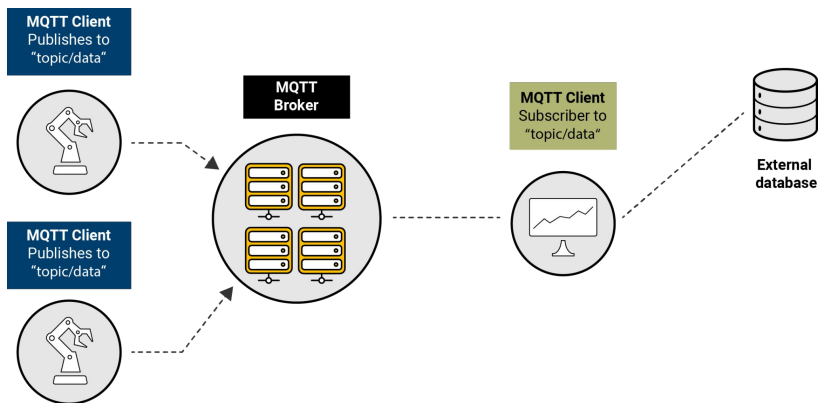
Group identifier

Actual topic

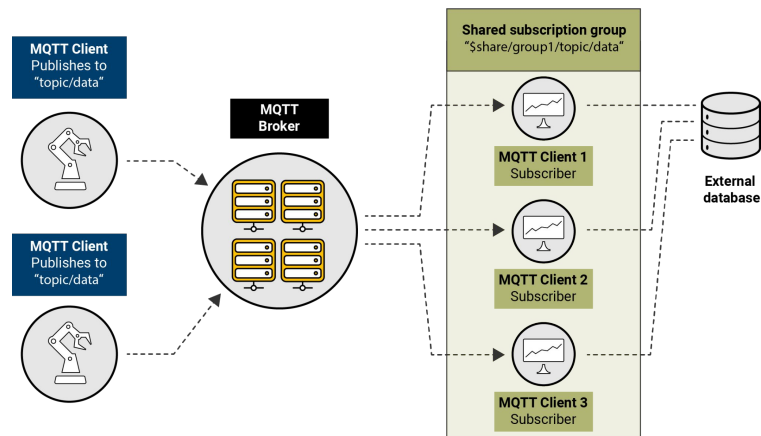


Shared Subscriptions' Example

Without Shared Subscription Group



With Shared Subscription Group



Benefits of Shared Subscriptions

- Facilitates scalable consumer infrastructure
- Provides good message throughput even for slow message processing situations
- High availability for consumers
- Enables really high volume messaging with shared consumer of messages



Vehicle Telemetry Persistence

- Connected cars
- Up to tens of millions of concurrently connected vehicles
- Individual topics for cars (VIN/telemetry)
- Shared wildcard topic for consumer (\$share/data-lake/+/telemetry)
- Persist information to single data source
- Enables central calculations (i.e. predictive maintenance)



Request - Response Pattern

Enables extensible, dynamic, and transparent implementation of “business acknowledgement” functionality

“Business acknowledgement” on top of regular given MQTT delivery guarantees

Pattern of communication that allows feedback to sender of messages

Still fully asynchronous

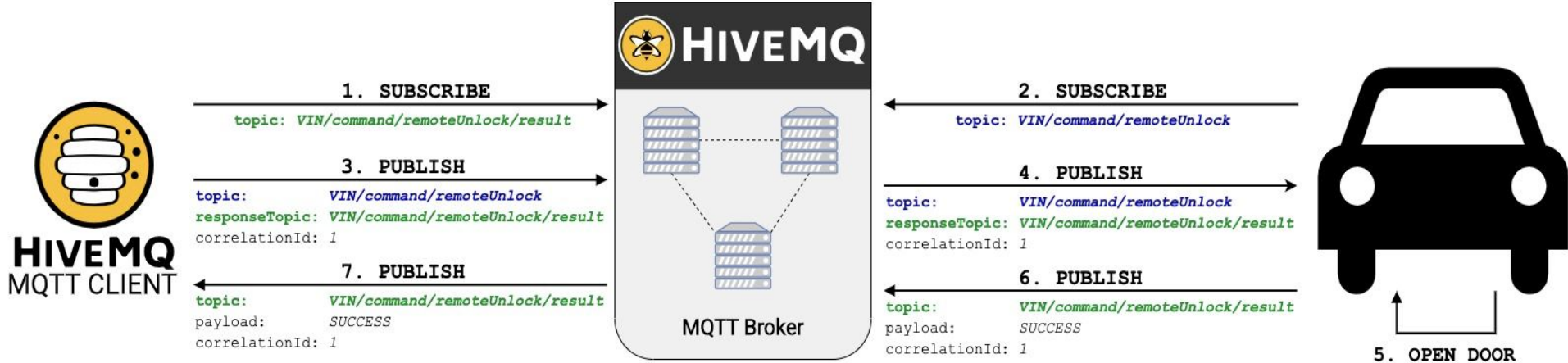


Benefits of Request - Response Pattern

- Enables use cases that require guaranteed delivery and processing of messages
 - Any time you are dealing with sensitive commands (For example, door lock/unlock, sending/receiving temperature, etc.)
- Provides a standardized framework for end-to-end acknowledgements
- Command audit



Request - Response Pattern

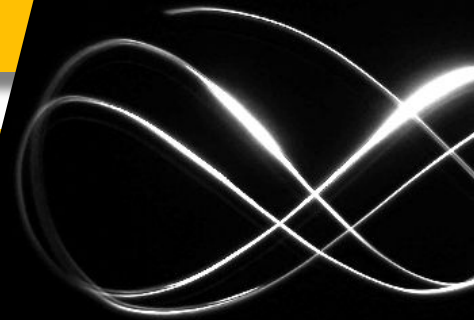


User Properties

Arbitrary number of Key-Value String pairs

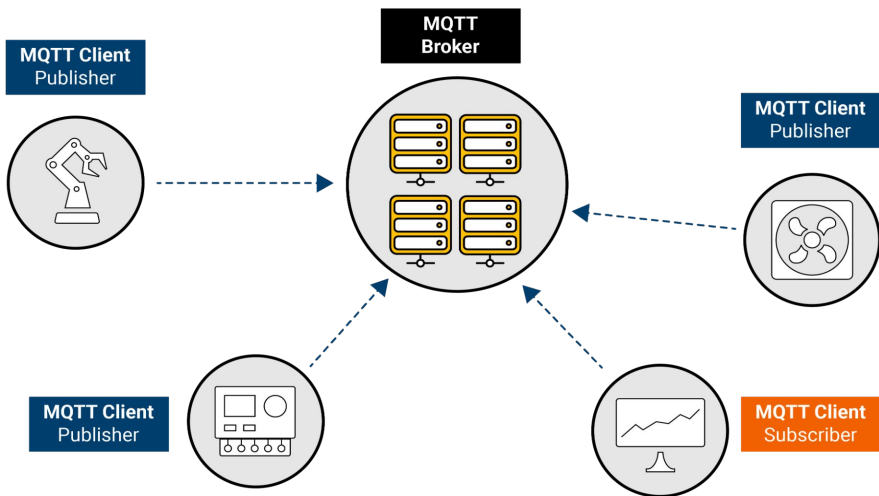
Provide near endless extensibility to most MQTT packets


Commonly known from other technologies like HTTP or Kafka



User Properties

Offers Limitless Possibilities to Apply Additional Logic



```
MQTT-Packet:
PUBLISH 
contains:                                     Example
packetId                                     4314
topicName                                     "topic/data"
qos                                           1
retainFlag                                    false
payload                                       "progress:58.4%"
dupFlag                                       false
userProperty                                 "route:storage"
```



Tracing

- Add message-id to PUBLISH packets
- End-to-end tracing of messages, potentially across multiple systems
 - Increases observability in IoT
 - The MQTT broker can be involved in identifying problems in surrounding systems



Session and Message Expiry

Time in seconds set by sending client (Publisher) for Message

Time in seconds set by client for Session

PUBLISH/Session expires when not used for a certain amount of time

Expires when Clients are not responding (PUBACK)

Expires when clients are offline



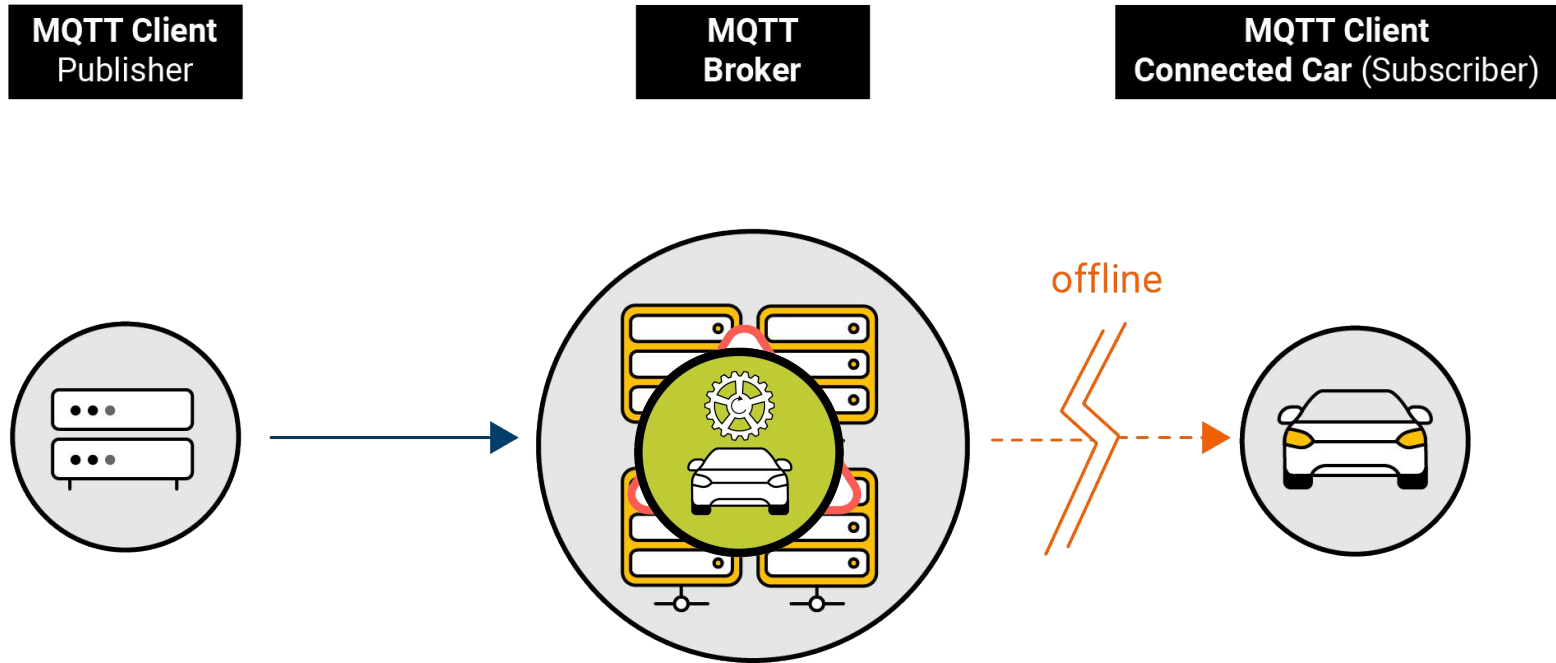
Benefits of Session and Message Expiry

- Allows users to assign different importance to different message types (message/service prioritization)
 - Delete some quickly
 - Keep others longer
- Housekeeping for decommissioned devices
 - If a device/client does not come back online for 14 days, remove all remnants of it from the broker
 - Saves infrastructure costs + potential security measure



Session and Message Expiry

Premium Navigation Service



Payload Type Description

Combination of 2 attributes in PUBLISH packet

Payload Format Indicator and Content Type

Binary or UTF-8 and arbitrary description for UTF-8 payloads



Payload Type Descriptors

Helps with standardization across verticals

MQTT-Packet:

PUBLISH



```
packetId                "client1"
topicName               "client1/status"
payloadFormatIndicator  1
contentType             SolarPanelSchemaV1.0
qos                    1
retainFlag              true
payload                 "online"
```

- 0 = binary byte stream
- 1 = UTF-8 encoded payload
- Content Type (if 1) => Describes the actual type of Payload (i.e. JSON Schema v3, XML)



Benefits of Payload Type Descriptors

- Support multiple standards across a single deployment
- Helps building cross-vendor type use cases
- IoT Solution provider for various customer solution
 - Version control for messaging
- Could use *User Properties* for binary payloads
- Enable different types of payloads in same use case



Smart Toll Booth

- Sending lists of license plates
- Transferring pictures
- Cannot differentiate by topic
- Payload descriptors ensure correct parsing and processing of each message



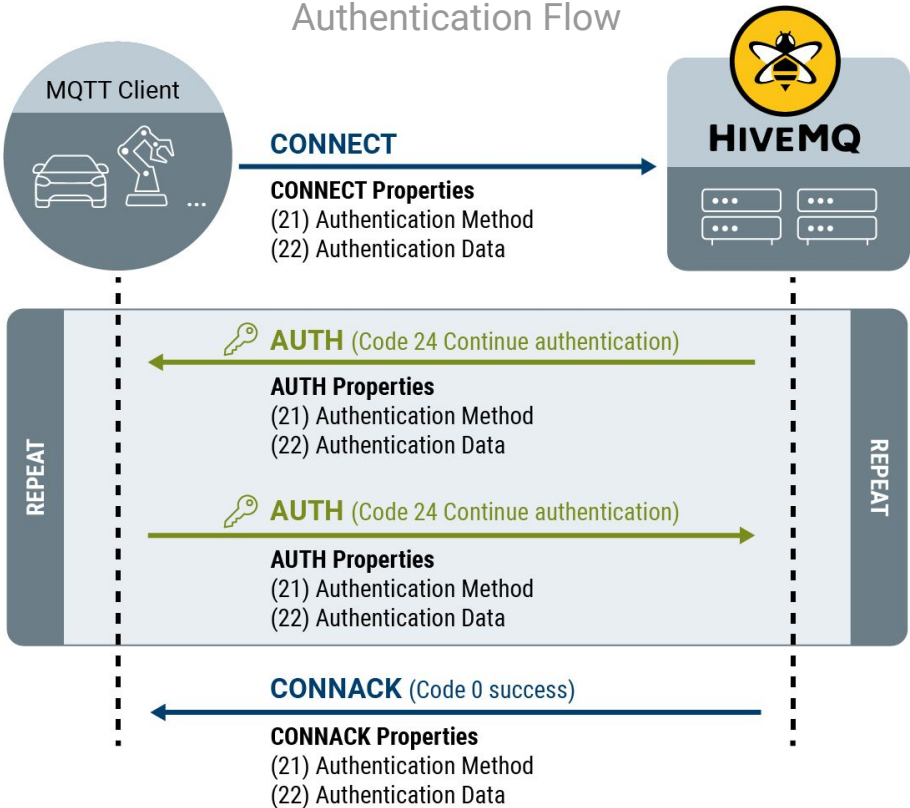
Enhanced Authentication

Additional packet introduced

Can be used for challenge-response authentication



Enhanced Authentication



Benefits of Enhanced Authentication

- Integrate modern IoT System into existing enterprise grade IT security infrastructure
 - SASL, Kerberos, etc.
 - Saves the need to introduce a separate system for your IoT devices



Resources

[MQTT 5 Essentials Series](#)

[MQTT 5 Video Series](#)

[Try HiveMQ Cloud](#)

[Download HiveMQ](#)



ANY QUESTIONS?


Reach out to community.hivemq.com




THANK YOU

Contact:

Florian Raschbichler
Head of Support at HiveMQ

 @fraschbi

 [linkedin.com/in/fraschbi/](https://www.linkedin.com/in/fraschbi/)

