WEBCAST

Revolutionizing IoT Testing: Sneak Peek of a New HiveMQ Tool

hosted by HIVEMQ

Speakers

Dominik Obermaier

HiveMQ CTO & Co-founder

dominik.obermaier@hivemq.com
linkedin.com/in/dobermai/
@dobermai

Georg Held

Engineering Manager @ HiveMQ

georg.held@hivemq.com
linkedin.com/in/sauroter/

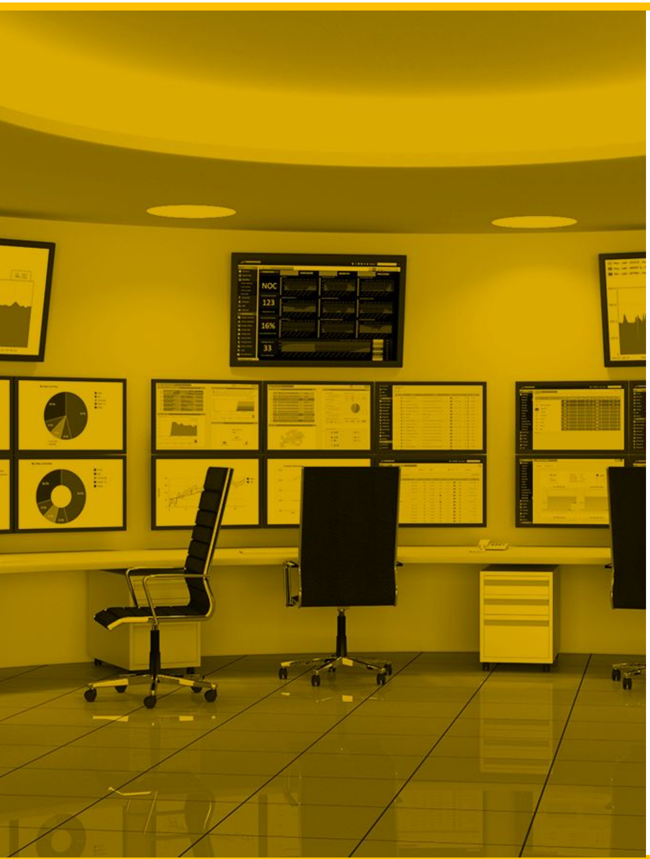
Why IoT Testing is Important

HIVEMQ

Why IoT Testing is Important

Fixing IoT Production Errors are Costly to Fix in the Field

Why IoT Testing is Important



Load & Stress Testing of Complete End-to-end IoT System is Required to Determine System Resilience

Why IoT Testing is Important?

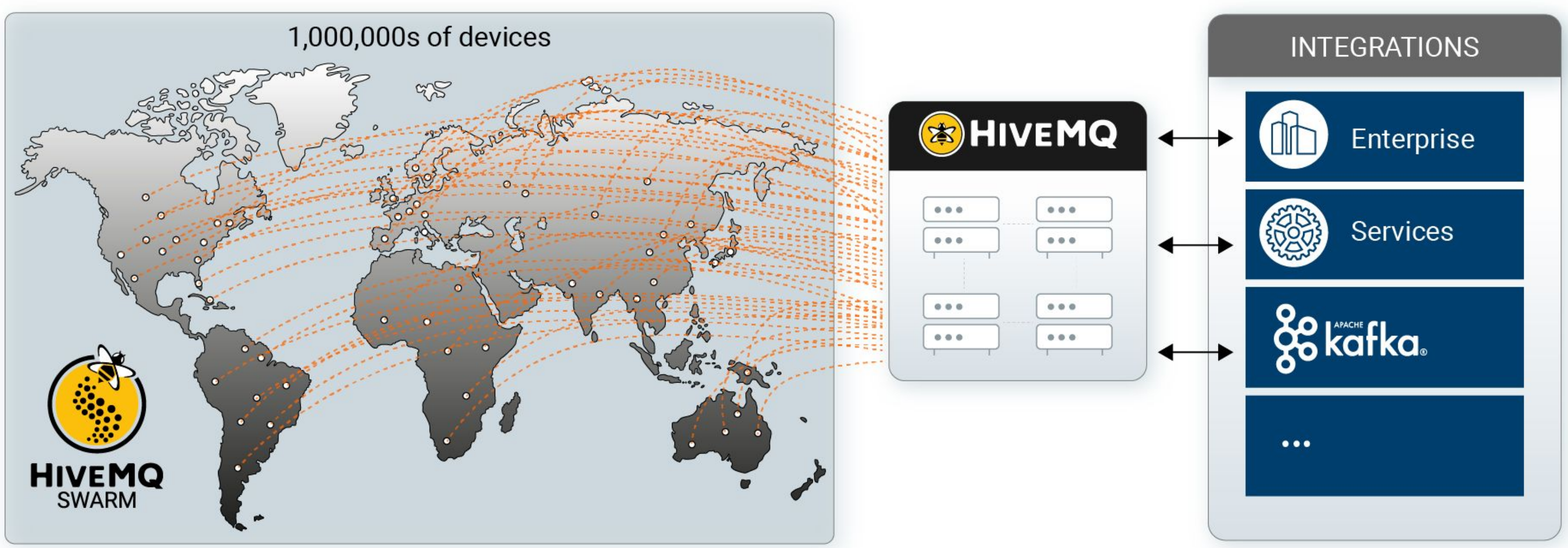
Capacity planning required to:

- Budget network and infrastructure costs
- Budget financial costing for cloud hosting

Challenges for IoT Testing

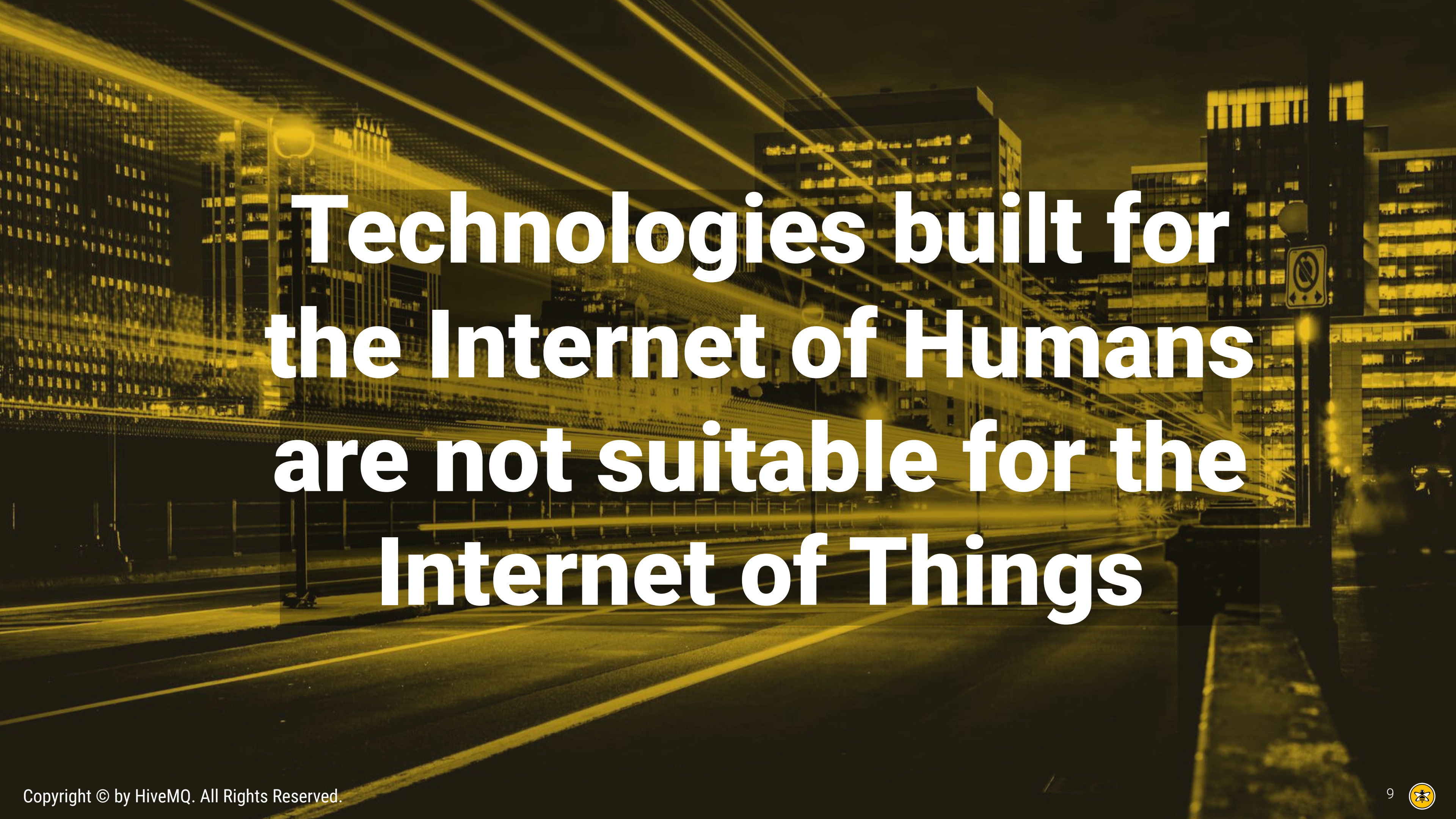
- IoT systems are massive distributed systems that can be difficult to test
- Test environment is often different from production behaviour
- Individual IoT devices can have multiple complex behaviour patterns
- IoT production data can have a high degree of variability
- Testing at massive scale

Challenges for IoT Testing



- Massive distribution
- Very hard to set up in testing / staging

- Single Systems (10s-100s)
- Easy to setup in testing /staging



Technologies built for the Internet of Humans are not suitable for the Internet of Things

Introducing HiveMQ Swarm

HIVEMQ
SWARM

- Distributed platform able to create millions of unique network connections
- Simulating millions of devices, messages and MQTT topics
- Develop reusable scenarios that simulate device behaviours
- Custom data generator that simulate complex use cases
- Resource friendly and easy deployment to public clouds (AWS, Azure, etc.) and Kubernetes

Use Cases

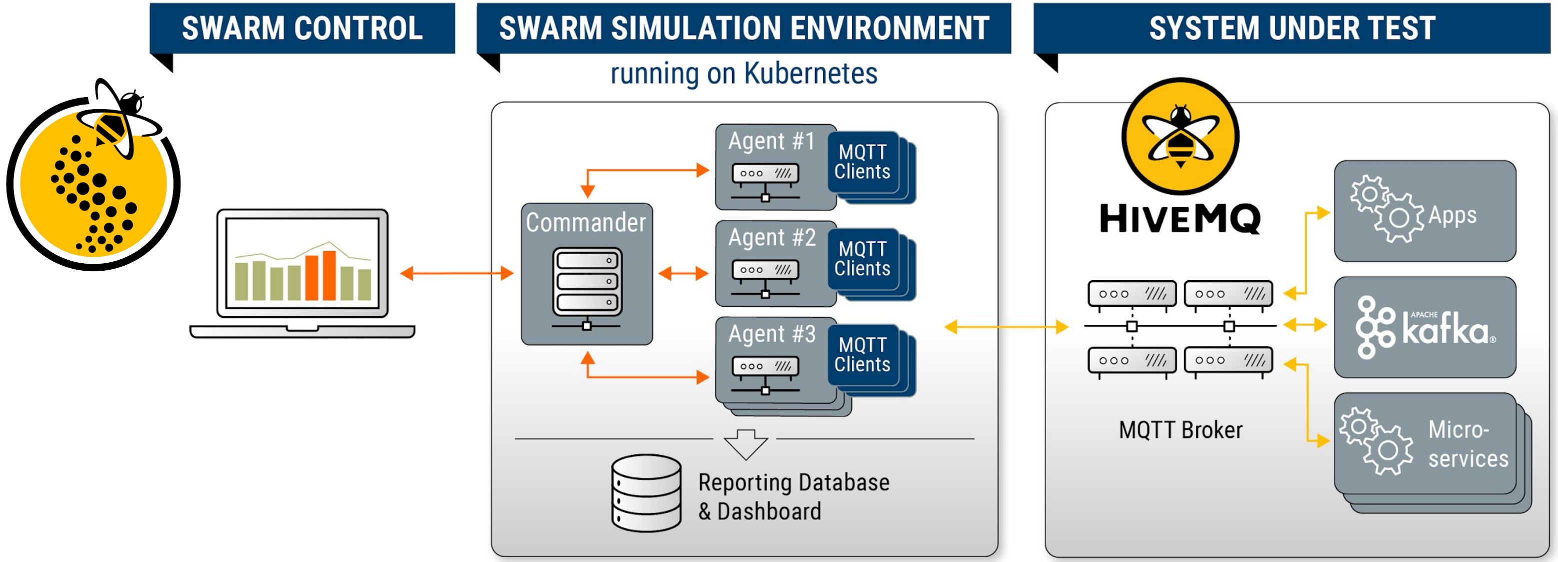
- Load and stress testing
- Capacity planning
- Quality assurance testing
- Test HiveMQ custom extensions
- IoT Scenario Testing
- Device rollout simulations
- Troubleshooting

HIVEMQ SWARM

Key Features

- Declarative and reusable scenarios
- Local and distributed setup
- Up to 10,000,000 real MQTT connections
- Built-in monitoring, logging, and reporting
- REST interface for metrics (Prometheus compatible)
- Custom data generator support (with SDK)
- Runs everywhere (Cloud, K8s, local DC, local machine)
- MQTT CLI integration

Distributed IoT Testing and Simulation



Swarm Lifecycle

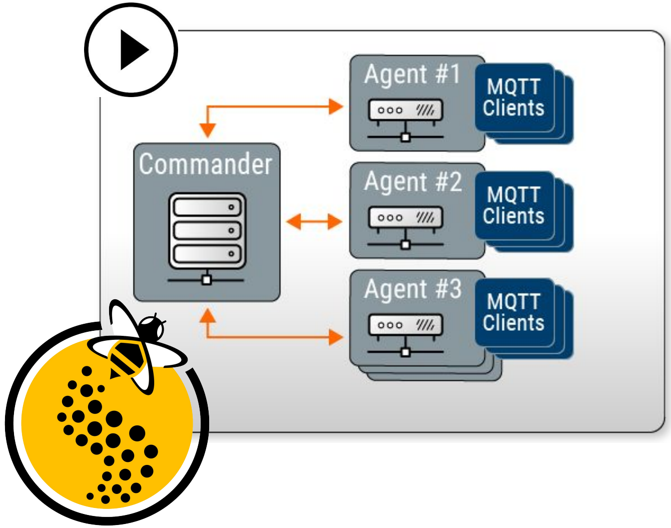
1. CREATE SCENARIO

```
<xml>
<brokers>
<address></address>

....
...

</xml>
```

2. EXECUTE IN SIMULATION ENVIRONMENT

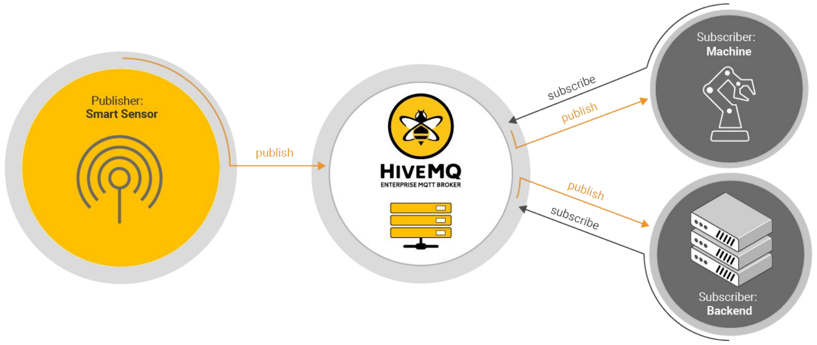


3. REPORT

REPEAT

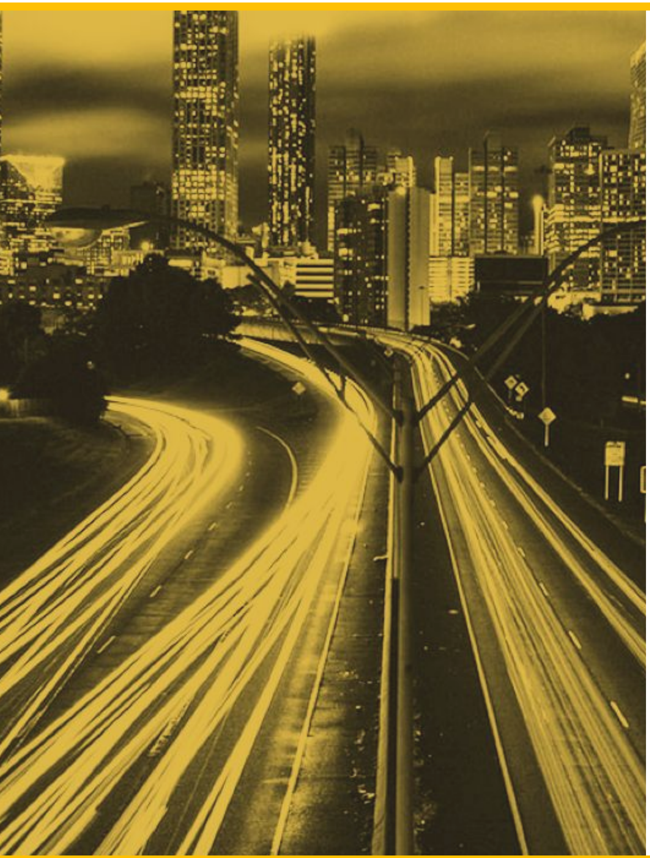
Refresher
MQTT

Introducing MQTT



- IoT messaging protocol
- Publish/subscribe
- Minimal overhead for client and bandwidth
- Designed for reliable communications over unreliable channels
- Efficient bi-directional messaging
- 3 Quality of Service (QoS) levels

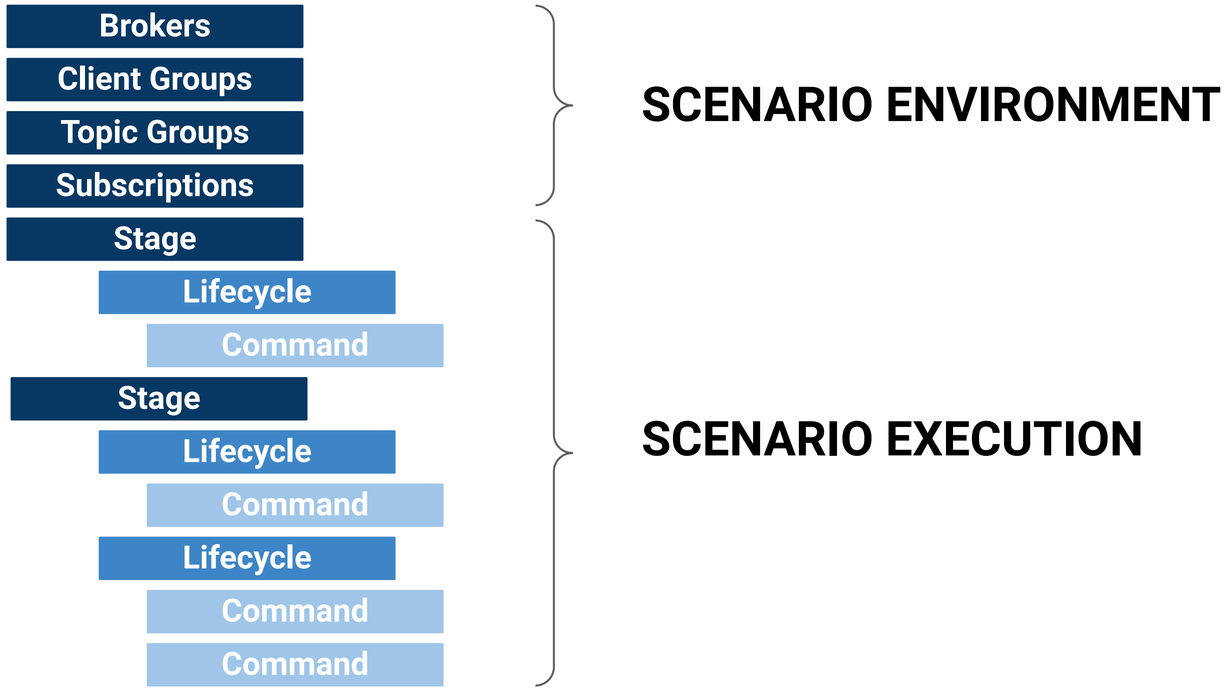
Benefits of MQTT



- Lightweight and efficient
- Bi-directional communications
- Scale to millions of things
- Reliable message delivery
- Support of unreliable networks
- Security Enabled

DEMO

The Scenario Structure



Scenario Environment

```xml
<brokers>
   <broker id="hivemq-cloud">
       <address>cloud.hivemq.com</address>
       <port>8883</port>
       <transport>TLS</transport>
   </broker>
</brokers>
<clientGroups>
   <clientGroup id="my-clients">
       <clientIdPattern>my-client-[0-9]{2}</clientIdPattern>
       <count>25</count>
   </clientGroup>
</clientGroups>
<topicGroups>
   <topicGroup id="my-topics">
       <topicNamePattern>topic/subtopic-[0-9]{2}</topicNamePattern>
       <count>10</count>
   </topicGroup>
</topicGroups>
```

Scenario Execution

```
<stages>
   <stage id="s1">
       <lifeCycle id="s1.l1" clientGroup="my-clients">
           <connect broker="hivemq-cloud" credentials="dXMzcg==:cGFzc3cwcmQ=" />
       </lifeCycle>
   </stage>
   <stage id="s2">
       <lifeCycle id="s2.l1" clientGroup="my-clients">
           <publish topicGroup="my-topics" payloadGeneratorType ="random message="1024"/>
           <disconnect/>
       </lifeCycle>
   </stage>
</stages>
```

Data Generators

- Real-live MQTT environments produce a multitude of semantic data:
  - PUBLISH payloads
  - Topic filters and Topics
  - Authentication and authorization information
  - Userproperties
  - …
- This data is on top of MQTT and encapsulates the business logic of the deployment.
- HiveMQ Swarm provides build in data generators and an SDK for custom generators.
- HiveMQ Swarm distributes and orchestrates the generated data across the test environment.

Data Generators Example: Payload

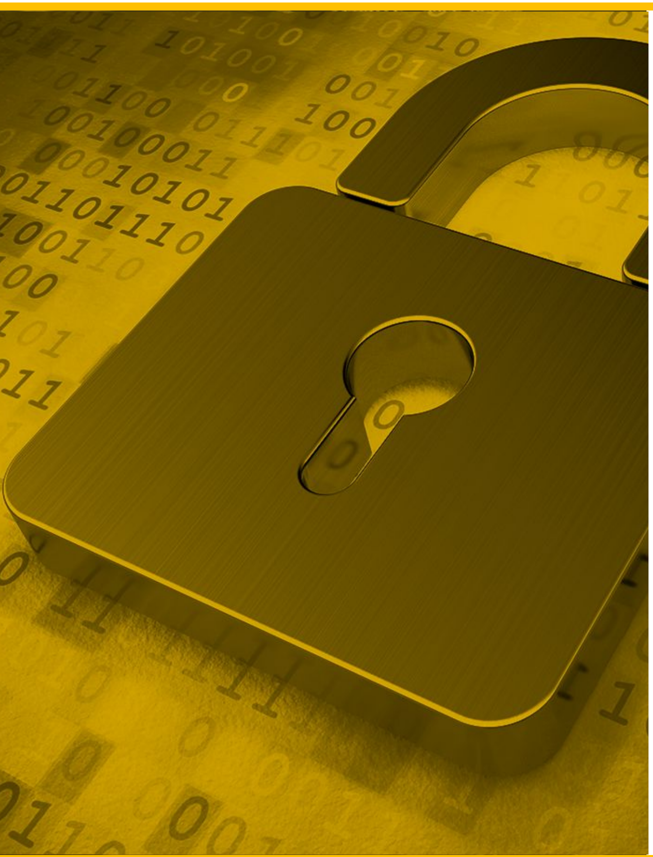
- Build-in: static, random, template-based, ...

```
<publish topicGroup="my-topics" payloadGeneratorType="random" message="1024"/>
```

- Custom, via the open source plugin SDK:

```
public class SparkplugProducer implements PayloadGenerator {
    @Override
    public @NotNull ByteBuffer nextPayload(
        final @NotNull PayloadGeneratorInput payloadGeneratorInput) {
          return sparkPlugTestData(payloadGeneratorInput);
    }
}
```

Security Providers



IoT security is a MUST

- Security systems are usually big, company specific, and difficult to interact with
- Security systems are already in place and not designed for IoT
- Security systems can be the bottleneck of an IoT deployment

HiveMQ Swarm enables testing of the entire IoT deployment, including the security systems.

Security Providers

HiveMQ Swarm comes with TLS support out of the box:

```
<broker id="hivemq-cloud" >
   <address>cloud.hivemq.com</ address>
   <port>8883</port>
   <transport>TLS</transport>
</broker>
```

The Standard Security Plugin provides basic authentication:

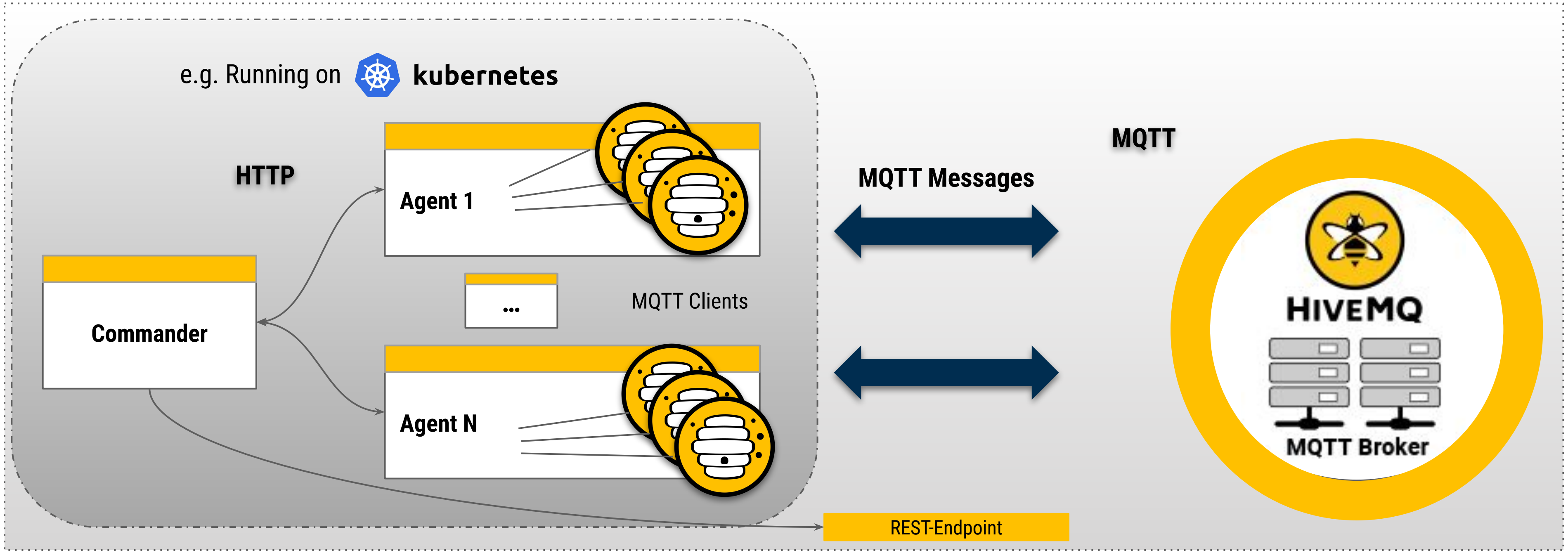
```
<connect broker="hivemq-cloud"  credentials="dXMzcg==:cGFzc3cwcmQ=" />
```

Custom Security Providers Example: OAuth

The plugin SDK can be used to integrate into every conceivable system:

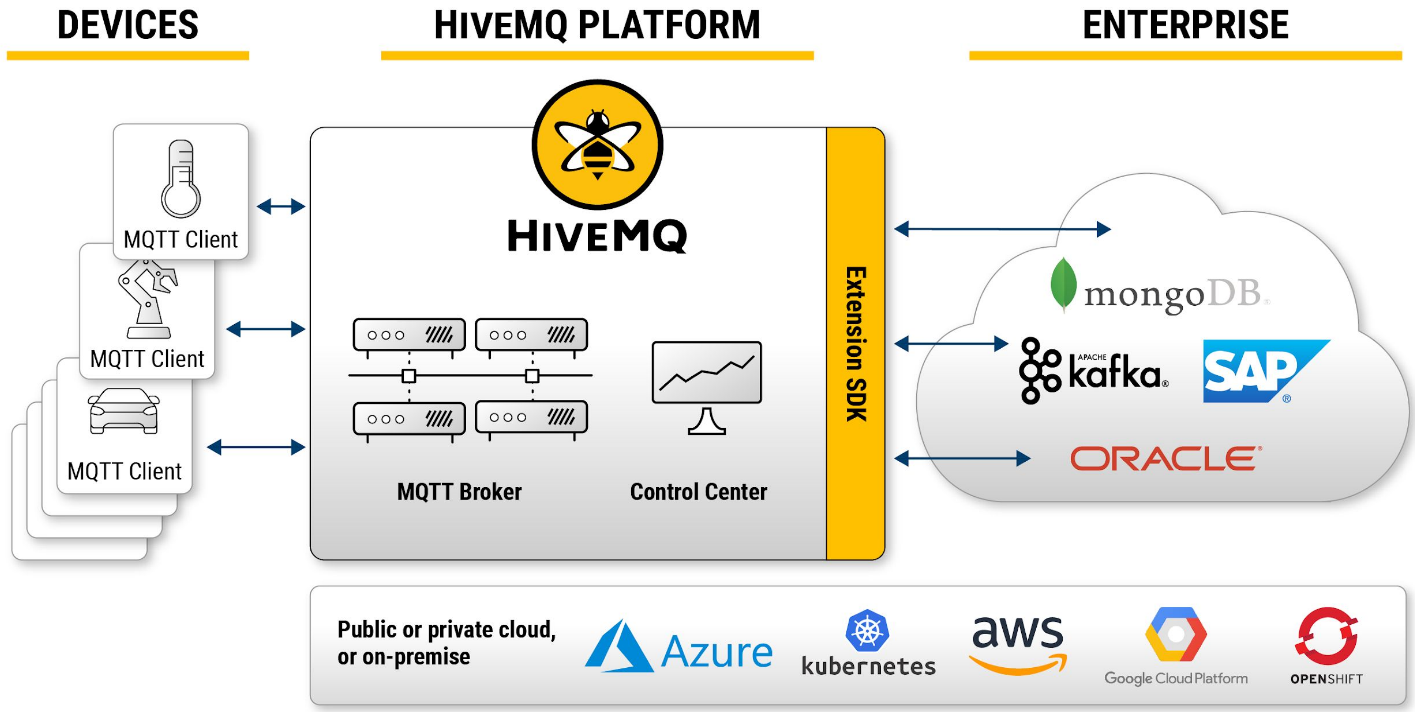
```
public class OAuthSecurity implements SecurityProvider {
    @Override
    public @NotNull Security provideSecurity(
            final @NotNull SecurityProviderInput input) {
        final byte[] jwt = OAuthService.oauthFlow(input);
        return Security.builder()
                .userNamePassword(input.getClientId(), jwt)
                .build();
    }
}
```

HiveMQ Swarm - Distributed Setup



Next Steps

Enterprise MQTT Platform

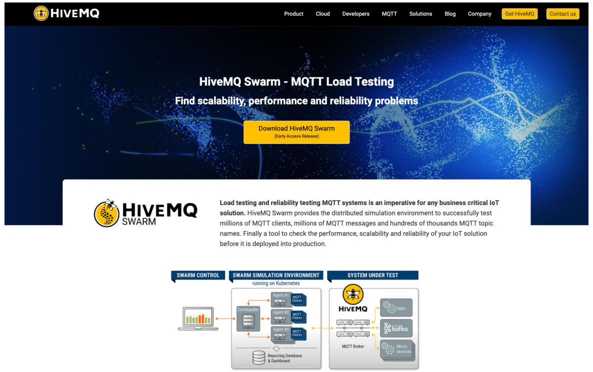


Next Steps

1. Scan the QR-Code

hivemq.com/hivemq-swarm

2. Visit the HiveMQ Swarm Page



HiveMQ Swarm - MQTT Load Testing

Find scalability, performance and reliability problems

Load testing and reliability testing MQTT systems is an imperative for any business critical IoT solution. HiveMQ Swarm provides the distributed simulation environment to successfully test millions of MQTT clients, millions of MQTT messages and hundreds of thousands MQTT topic names. Finally a tool to check the performance, scalability and reliability of your IoT solution before it is deployed into production.

3. Download HiveMQ Swarm Early Access today

HiveMQ Platform

HIVEMQ BRIDGE EXTENSION

HIVEMQ SWARM

HIVEMQ ENTERPRISE EXTENSION FOR KAFKA

MQTT CLI

HIVEMQ Enterprise Security Extension

HIVEMQ K8s OPERATOR

HIVEMQ ENTERPRISE

HiveMQ Portfolio

Broker	HIVEMQ COMMUNITY	HIVEMQ PROFESSIONAL	HIVEMQ ENTERPRISE	HIVEMQ CLOUD			
Clients	HIVEMQ MQTT CLIENT						
Enterprise Extensions	HIVEMQ ENTERPRISE EXTENSION FOR KAFKA	HIVEMQ Enterprise Security Extension	HIVEMQ ENTERPRISE BRIDGE EXTENSION				
Tools & Ecosystem	HIVEMQ TESTCONTAINER	MQTT CLI	HIVEMQ K8s OPERATOR	HiveMQ Docker Images	HiveMQ AMI	Swarm (bitte links)	Mosquitto To HiveMQ

ANY QUESTIONS?

Reach out to community.hivemq.com

THANK YOU

Contact Details

Dominik Obermaier

dominik.obermaier@hivemq.com

linkedin.com/in/dobermai/

@dobermai

Georg Held

georg.held@hivemq.com

linkedin.com/in/sauroter/

HIVEMQ