Varroa

MQTT-Scenario-Testing-Tool

R. Atherton, S. Baier, S. Giebl, G. Held, Y. Weber, T. Weiden

Contents

1 Vision

The Name of our MQTT-Testing-Tool (Varroa) is inspired by the varroa mite, which is a species of mite that infects honey bee colonies. This name has been chosen due to it working in a similar way but instead of infesting a hive, it tries to infest a broker. The inspiration for this name came from the broker 'HiveMQ' and it's branding. The basic use-case of Varroa is testing the resilience of brokers by creating load. Hereby load is defined by a number of MQTT-clients sending different sequences of MQTT-messages to the broker. Which sequences get carried out in which order is determined by a Scenario. A scenario defines the temporal execution as well as the amount of actions across a MQTT-network and the topology of the network. The motivation for the creation of this project was the lack of testability of MQTT-systems.

Varroa is organized as a distributed system, due to the impossibility of creating enough MQTT-clients on a single machine to overload a MQTT-broker, especially if the broker is also a distributed system.

```xml
<?xml version="1.0" encoding="utf-8"?>
<xs:schema attributeFormDefault="unqualified" elementFormDefault="quali
    xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="points">
    <xs:complexType>
      <xs:sequence>
        <xs:element maxOccurs="unbounded" name="point">
          <xs:complexType>
            <xs:attribute name="x" type="xs:unsignedShort" use="require
            <xs:attribute name="y" type="xs:unsignedShort" use="require
          </xs:complexType>
        </xs:element>
      </xs:sequence>
```

```
      </xs:complexType>
    </xs:element>
</xs:schema>
```

2 Concepts

To understand the workings of Varroa, we will have to take a look at the different parts that make up the system.

2.1 Definitions

Commander   Part of the Command and Control structure, that parses the scenario, generates chunks and distributes them to the Agents.

Agent   The Agent is part of the Command and Control structure. He receives Chunks from the Commander and passes them to the MQTT-Agents.

Varroa Instance   A running Varroa process in a single JVM, can be either Commander or Agent.

MQTT Agent   Part of the load structure, that takes chunks and executes them with MqttBee clients.

MQTT Client   A MqttBee client used to execute the Commands that are defined in the Chunks.

Client group   A group of Clients that share similar behaviour and properties.

Command

#	Title	User Story	Importance
1	Transparency	Varroa has to be comprehensible for the user.	Must have high
2	10.000.000 MQTT Clients	Varroa has to be able to generate a large amount of clients.	Must have high
3	Scalability	Varroa should scale vertically with relatively low scaling costs.	Must have
4	Determinism	Varroa has to work in deterministic ways, meaning it should produce the same result for a Scenario every time.	Must have
5	Distributed	Varroa is a distributed System.	Must have low
6	Usabillity	Varroa has to be easily usable.	Very important
7	Code Quality	Varroa's coding quality should be very high.	Important
8	Stability	Varroa has to run in a stable manner.	Important
9	Resource efficiency	Varroa has to use the available computation and memory resources efficiently.	Important
10	User / Developer Guide	Varroa needs a User / Developer Guide.	Somewhat important
11	Automation capacity	Varroa should be automatable	Somewhat important

Chunk The scenario is split in Chunks by the Commander and then those Chunk are distributed to the agents.

Scenario A scenario is an abstract representation of a real MQTT-Use-Case. It defines the topology of all participating MQTT clients and brokers. The scenario enables the simulation of a large amount of MQTT clients.

2.2 Requirements

List of Figures

List of Tables