MQTT使用说明

**版本 1.0.1**

**浩鲸云计算科技股份有限公司**

## MQTT简介

MQTT（Message Queuing Telemetry Transport，消息队列遥测传输）是IBM开发的一个即时通讯协议，有可能成为物联网的重要组成部分。该协议支持所有平台，几乎可以把所有联网物品和外部连接起来，被用来当做传感器和制动器（比如通过Twitter让房屋联网）的通信协议。

早在1999年，IBM的Andy Stanford-Clark博士以及Arcom公司ArlenNipper博士发明了MQTT（Message Queuing Telemetry Transport，消息队列遥测传输）技术  。

## MQTT特点

MQTT协议是为大量**计算能力有限**，且工作在**低带宽、不可靠的网络的远程传感器和控制设备通讯**而设计的协议，它具有以下主要的几项特性：

1、使用发布/订阅消息模式，提供一对多的消息发布，解除应用程序耦合；

2、对负载内容屏蔽的消息传输；

3、使用 TCP/IP 提供网络连接；

4、有三种消息发布服务质量：

“**至多一次**”，消息发布完全依赖底层 TCP/IP 网络。会发生消息丢失或重复。这一级别可用于如下情况，环境传感器数据，丢失一次读记录无所谓，因为不久后还会有第二次发送。

“**至少一次**”，确保消息到达，但消息重复可能会发生。

“**只有一次**”，确保消息到达一次。这一级别可用于如下情况，在计费系统中，消息重复或丢失会导致不正确的结果。

5、小型传输，开销很小（固定长度的头部是 2 字节），协议交换最小化，以降低网络流量；

6、使用 Last Will 和 Testament 特性通知有关各方客户端异常中断的机制；

## 实现方式

实现MQTT协议需要客户端和服务器端通讯完成，在通讯过程中，MQTT协议中有三种身份：发布者（Publish）、代理（Broker）（服务器）、订阅者（Subscribe）。其中，消息的发布者和订阅者都是客户端，消息代理是服务器，消息发布者可以同时是订阅者。

MQTT传输的消息分为：主题（Topic）和负载（payload）两部分：

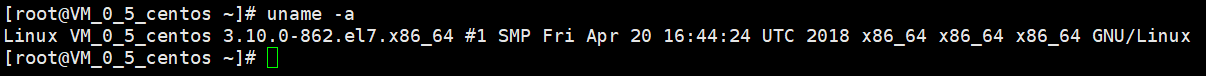
　　（1）Topic，可以理解为消息的类型，订阅者订阅（Subscribe）后，就会收到该主题的消息内容（payload）；

　　（2）payload，可以理解为消息的内容，是指订阅者具体要使用的内容。

## 软件安装及配置

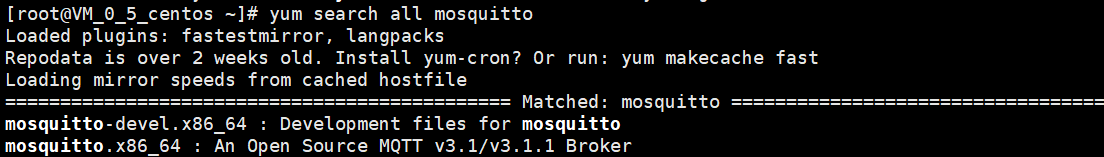
### 软件安装

安装环境：CentOS\_7.2



我们使用yum安装mosquito

yum search all mosquitto

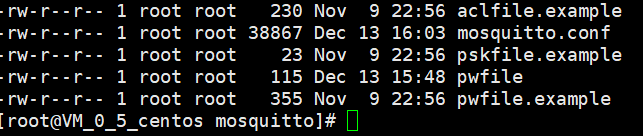


版本：mosquitto v3.1/v3.1.1

yum install mosquitto

## 配置

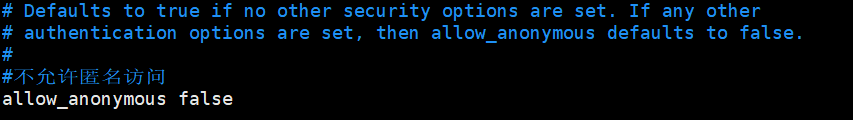
安装完成之后，所有配置文件会被放置于/etc/mosquitto/目录下



我们主要来配置mosquitto.conf文件

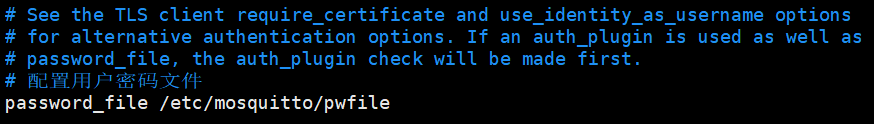
将配置文件修改为：  
#不允许匿名访问

allow\_anonymous false



# 配置用户密码文件

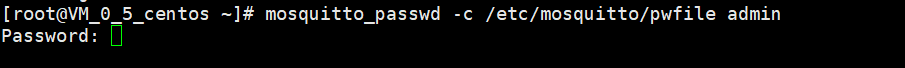
password\_file /etc/mosquitto/pwfile



* **添加用户信息**

**添加用户：‘admin’ 密码：‘admin’**

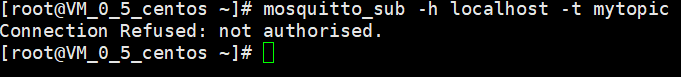
**mosquitto\_passwd -c /etc/mosquitto/pwfile admin**



* **用户认证测试**

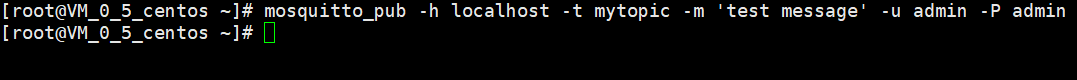
**重启Mosquitto**

**未使用密码时，我们尝试连接。结果提示拒绝连接.**

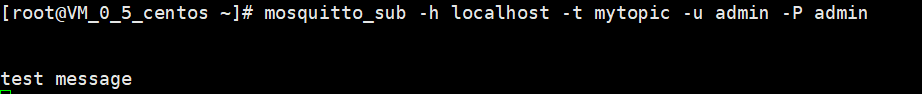


**输入正确用户密码.**

**我们先来发布一条消息.**



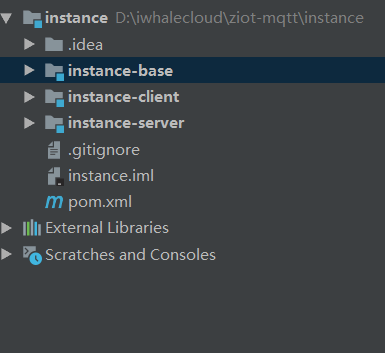
**此时可以看到消费者已经成功消费了一条消息了!**



**配置文件其他例如**客户端心跳的间隔时间、客户端心跳消息的最大并发数、客户端心跳消息缓存队列等使用的默认配置，我们可根据业务需求及实际情况进行修改！

## 代码实现

本项目主要实现mqtt服务器的发布订阅通信功能。订阅者客户端先订阅主题Topic等待接收到消息，再通过接口调用发布者客户端，发布消息到指定的Topic。最后订阅者成功接收到消息。项目目录结构如下：



instance-base模块的pom文件主要添加mqtt和slf4j的依赖包。

<!-- mqtt -->  
<dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-integration</artifactId>  
</dependency>  
<dependency>  
 <groupId>org.springframework.integration</groupId>  
 <artifactId>spring-integration-stream</artifactId>  
</dependency>  
<dependency>  
 <groupId>org.springframework.integration</groupId>  
 <artifactId>spring-integration-mqtt</artifactId>  
</dependency>  
<!-- slf4j -->  
<dependency>  
 <groupId>org.slf4j</groupId>  
 <artifactId>slf4j-api</artifactId>  
 <version>1.7.25</version>  
</dependency>

* 消息订阅者：instance-client
* 消息发布者：instance-server

#### MQTT配置信息如下：

#MQTT配置信息  
#MQTT-用户名  
spring.mqtt.username=admin  
#MQTT-密码  
spring.mqtt.password=admin  
#MQTT-服务器连接地址，如果有多个，用逗号隔开，如：tcp://127.0.0.1:1883，tcp://192.168.2.133:1883  
spring.mqtt.url=tcp://www.zhhcloud.cn:1883  
#MQTT-连接服务器默认客户端ID  
spring.mqtt.client.id=mqttId  
#MQTT-默认的消息推送主题，实际可在调用接口时指定  
spring.mqtt.default.topic=topic  
#消息质量  
spring.mqtt.qos=1  
#连接超时  
spring.mqtt.completionTimeout=10000

#### 订阅者配置类MqttClientConfig：

*/\*\*  
 \* Copyright (C), 2018-2018  
 \* FileName: MqttSenderConfig  
 \* Author: LoadHao  
 \* Date: 2018/12/13 14:30  
 \* Description: mqtt 发送消息端配置  
 \* History:  
 \* <author> <time> <version> <desc>  
 \* 作者姓名 修改时间 版本号 描述  
 \*/*package com.iwhalecloud.mqtt.instance.client.mqtt;  
  
import lombok.extern.slf4j.Slf4j;  
import org.eclipse.paho.client.mqttv3.MqttConnectOptions;  
import org.springframework.beans.factory.annotation.Value;  
import org.springframework.boot.autoconfigure.liquibase.LiquibaseDataSource;  
import org.springframework.boot.json.JsonParser;  
import org.springframework.context.annotation.Bean;  
import org.springframework.context.annotation.Configuration;  
import org.springframework.integration.annotation.IntegrationComponentScan;  
import org.springframework.integration.annotation.ServiceActivator;  
import org.springframework.integration.channel.DirectChannel;  
import org.springframework.integration.core.MessageProducer;  
import org.springframework.integration.dsl.IntegrationFlow;  
import org.springframework.integration.dsl.IntegrationFlows;  
import org.springframework.integration.mqtt.core.DefaultMqttPahoClientFactory;  
import org.springframework.integration.mqtt.core.MqttPahoClientFactory;  
import org.springframework.integration.mqtt.inbound.MqttPahoMessageDrivenChannelAdapter;  
import org.springframework.integration.mqtt.outbound.MqttPahoMessageHandler;  
import org.springframework.integration.mqtt.support.DefaultPahoMessageConverter;  
import org.springframework.messaging.Message;  
import org.springframework.messaging.MessageChannel;  
import org.springframework.messaging.MessageHandler;  
import org.springframework.messaging.MessagingException;  
  
*/\*\*  
 \* 功能描述: <br>  
 \* 〈mqtt 发送消息端配置〉  
 \*  
 \** ***@author*** *LoadHao  
 \** ***@create*** *2018/12/13  
 \** ***@since*** *1.0.0  
 \*/*@Slf4j  
@Configuration  
@IntegrationComponentScan  
public class MqttClientConfig {  
 @Value("${spring.mqtt.username}")  
 private String username;  
  
 @Value("${spring.mqtt.password}")  
 private String password;  
  
 @Value("${spring.mqtt.url}")  
 private String hostUrl;  
  
 @Value("${spring.mqtt.client.id}")  
 private String clientId;  
  
 @Value("${spring.mqtt.default.topic}")  
 private String defaultTopic;  
  
 @Value("${spring.mqtt.qos}")  
 private int qos;  
  
 @Value("${spring.mqtt.completionTimeout}")  
 private int completionTimeout;  
  
 @Bean  
 public MqttConnectOptions getMqttConnectOptions() {  
 MqttConnectOptions mqttConnectOptions = new MqttConnectOptions();  
 mqttConnectOptions.setUserName(username);  
 mqttConnectOptions.setPassword(password.toCharArray());  
 mqttConnectOptions.setServerURIs(new String[]{hostUrl});  
 mqttConnectOptions.setKeepAliveInterval(2);  
 return mqttConnectOptions;  
 }  
  
 @Bean  
 public MqttPahoClientFactory mqttClientFactory() {  
 DefaultMqttPahoClientFactory factory = new DefaultMqttPahoClientFactory();  
 factory.setConnectionOptions(getMqttConnectOptions());  
 return factory;  
 }  
  
 @Bean  
 public MessageChannel mqttInputChannel() {  
 return new DirectChannel();  
 }  
  
 @Bean  
 public MessageProducer inbound() {  
 MqttPahoMessageDrivenChannelAdapter adapter = new MqttPahoMessageDrivenChannelAdapter(clientId, mqttClientFactory(), defaultTopic);  
 adapter.setCompletionTimeout(completionTimeout);  
 adapter.setConverter(new DefaultPahoMessageConverter());  
 adapter.setQos(qos);  
 adapter.setOutputChannel(mqttInputChannel());  
 return adapter;  
 }  
  
 @Bean  
 @ServiceActivator(inputChannel = "mqttInputChannel")  
 public MessageHandler handler() {  
 return message -> {  
 try {  
 //这里拿到发布的消息内容，做具体的业务逻辑处理  
 String string = message.getPayload().toString();  
 *log*.info("Message :" + string);  
 } catch (MessagingException ex) {  
 *log*.error("MassageHandler is error!++++" + ex + "++++");  
 }  
 };  
 }  
}

#### 发布者配置类MqttServerConfig：

*/\*\*  
 \* Copyright (C), 2018-2018  
 \* FileName: MqttSenderConfig  
 \* Author: LoadHao  
 \* Date: 2018/12/13 14:30  
 \* Description: mqtt 发送消息端配置  
 \* History:  
 \* <author> <time> <version> <desc>  
 \* 作者姓名 修改时间 版本号 描述  
 \*/*package com.iwhalecloud.mqtt.instance.server.mqtt;  
  
import org.eclipse.paho.client.mqttv3.MqttConnectOptions;  
import org.springframework.beans.factory.annotation.Value;  
import org.springframework.context.annotation.Bean;  
import org.springframework.context.annotation.Configuration;  
import org.springframework.integration.annotation.IntegrationComponentScan;  
import org.springframework.integration.annotation.ServiceActivator;  
import org.springframework.integration.channel.DirectChannel;  
import org.springframework.integration.mqtt.core.DefaultMqttPahoClientFactory;  
import org.springframework.integration.mqtt.core.MqttPahoClientFactory;  
import org.springframework.integration.mqtt.outbound.MqttPahoMessageHandler;  
import org.springframework.messaging.MessageChannel;  
import org.springframework.messaging.MessageHandler;  
  
*/\*\*  
 \* 功能描述: <br>  
 \* 〈mqtt 发送消息端配置〉  
 \*  
 \** ***@author*** *LoadHao  
 \** ***@create*** *2018/12/13  
 \** ***@since*** *1.0.0  
 \*/*@Configuration  
@IntegrationComponentScan  
public class MqttServerConfig {  
 @Value("${spring.mqtt.username}")  
 private String username;  
  
 @Value("${spring.mqtt.password}")  
 private String password;  
  
 @Value("${spring.mqtt.url}")  
 private String hostUrl;  
  
 @Value("${spring.mqtt.service.id}")  
 private String serviceId;  
  
 @Value("${spring.mqtt.default.topic}")  
 private String defaultTopic;  
  
 @Bean  
 public MqttConnectOptions getMqttConnectOptions() {  
 MqttConnectOptions mqttConnectOptions = new MqttConnectOptions();  
 mqttConnectOptions.setUserName(username);  
 mqttConnectOptions.setPassword(password.toCharArray());  
 mqttConnectOptions.setServerURIs(new String[]{hostUrl});  
 mqttConnectOptions.setKeepAliveInterval(2);  
 return mqttConnectOptions;  
 }  
  
 @Bean  
 public MqttPahoClientFactory mqttClientFactory() {  
 DefaultMqttPahoClientFactory factory = new DefaultMqttPahoClientFactory();  
 factory.setConnectionOptions(getMqttConnectOptions());  
 return factory;  
 }  
  
 @Bean  
 @ServiceActivator(inputChannel = "mqttOutboundChannel")  
 public MessageHandler mqttOutbound() {  
 MqttPahoMessageHandler messageHandler = new MqttPahoMessageHandler(serviceId, mqttClientFactory());  
 messageHandler.setAsync(true);  
 messageHandler.setDefaultTopic(defaultTopic);  
 return messageHandler;  
 }  
  
 @Bean  
 public MessageChannel mqttOutboundChannel() {  
 return new DirectChannel();  
 }  
}

#### 消息推送接口

*/\*\*  
 \* Copyright (C), 2018-2018  
 \* FileName: MqttGateway  
 \* Author: LoadHao  
 \* Date: 2018/12/13 14:33  
 \* Description: 消息推送接口类  
 \* History:  
 \* <author> <time> <version> <desc>  
 \* 作者姓名 修改时间 版本号 描述  
 \*/*package com.iwhalecloud.mqtt.instance.server.mqtt;  
  
import org.springframework.integration.annotation.MessagingGateway;  
import org.springframework.integration.mqtt.support.MqttHeaders;  
import org.springframework.messaging.handler.annotation.Header;  
  
*/\*\*  
 \* 功能描述: <br>  
 \* 〈消息推送接口〉  
 \*  
 \** ***@author*** *LoadHao  
 \** ***@create*** *2018/12/13  
 \** ***@since*** *1.0.0  
 \*/*@MessagingGateway(defaultRequestChannel = "mqttOutboundChannel")  
public interface MqttGateway {  
  
 void sendToMqtt(String data);  
  
 void sendToMqtt(@Header(MqttHeaders.*TOPIC*) String topic, String payload);  
  
 void sendToMqtt(@Header(MqttHeaders.*TOPIC*) String topic, @Header(MqttHeaders.*QOS*) int qos, String payload);  
}

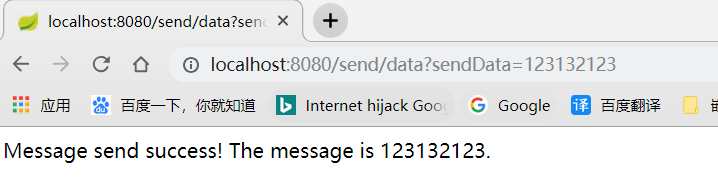
至此项目消息订阅者和发布者已经实现.下面我们来写一个测试接口.

#### 测试接口如下：

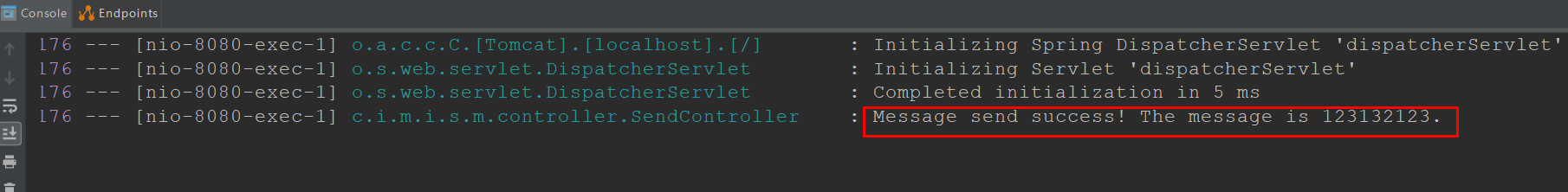


先启动instance-client，再启动instance-server.

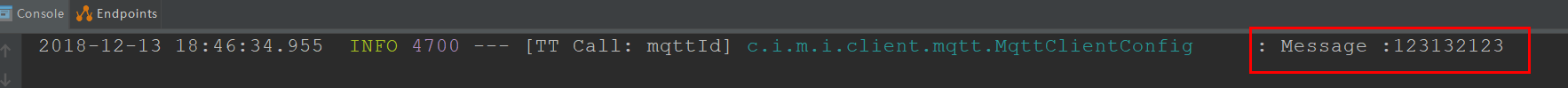
浏览器请求测试接口，并传递消息参数. 响应结果如下：



消息发布者打印信息：



消息订阅者打印信息：



可以看到消息已经成功被订阅者消费！

测试项目地址：<https://github.com/Loadhao/ziot-mqtt>

完结！