# MQTT调研测试报告

## 背景：

在物联网领域逐渐复杂化的网络通讯环境中，mqtt协议担当者非常重要的角色，主流语言关于mqtt的实现也非常之多！另外，公司目前使用6台服务器部署mosca，4台服务器部署emqtt，总共10台服务器用于mqtt server的搭建。 因此，为了挑选出性能、稳定、低耗且易二次开发的mqtt开源产品，我们全方位的对各个mqtt server实现做了调研测试，选出为最合适的mqtt server，节省服务器为全公司提供支撑服务！

## 调研对象：

本次供调研了9个mqtt的开源实现，涉及语言有Golang、Java、Erlang、Node、C，各相关地址为：

1. Golang: <https://github.com/VolantMQ/volantmq>
2. Golang: <https://github.com/zentures/surgemq>
3. Golang: <https://github.com/emitter-io/emitter>
4. Golang: <https://github.com/mafanr/meq>
5. Java：<http://activemq.apache.org/mqtt.html>
6. Java：<http://www.rabbitmq.com/mqtt.html>
7. Erlang：<https://github.com/emqx/emqx>
8. Node：<https://github.com/mcollina/mosca>
9. C：<https://github.com/eclipse/mosquitto>

## 调研：

调研仅通过社区的一些必要因素做对比，社区贡献者活跃度低的将会在此阶段被过滤掉（Rabbit没有数据可以参考，不参与一下对比）

## 性能测试

volantmq文档稀缺且关注量极少，这里不做考虑，surgemq社区不活跃，meq的贡献值只有一人，综合下来，golang中我们选择了emitter。另外，在Java领域挑选的activemq、rabbitmq两个项目也不考虑（性能和内存相比golang差距很大）。接下来，将会对emitter、emqtt、mosca(Max QoS1)、mosquitto进行性能测试！

### 测试环境

#### 系统环境

* CPU：Intel(R) Core(TM) i7-8700K CPU @ 3.70GHz
* OS：centos-release-7-6.1810.2.el7.centos.x86\_64
* 内核版本：Linux localhost.localdomain 3.10.0-957.1.3.el7.x86\_64 #1 SMP Thu Nov 29 14:49:43 UTC 2018 x86\_64 x86\_64 x86\_64 GNU/Linux
* 一核4G内存 HDD

#### TCP内核参数

net.ipv4.conf.default.arp\_announce = 2

net.ipv4.conf.lo.arp\_announce=2

net.ipv4.conf.all.arp\_announce=2

net.ipv4.tcp\_max\_tw\_buckets = 5000

net.ipv4.tcp\_syncookies = 1

net.ipv4.tcp\_max\_syn\_backlog = 1024

net.ipv4.tcp\_synack\_retries = 2

net.ipv6.conf.all.disable\_ipv6 = 1

net.ipv6.conf.default.disable\_ipv6 = 1

net.ipv6.conf.lo.disable\_ipv6 = 1

fs.file-max=65535

fs.inotify.max\_user\_instances = 8192

net.ipv4.tcp\_syncookies = 1

net.ipv4.tcp\_fin\_timeout = 30

net.ipv4.tcp\_tw\_reuse = 1

net.ipv4.ip\_local\_port\_range = 1024 65000

net.ipv4.tcp\_max\_syn\_backlog = 65536

net.ipv4.tcp\_max\_tw\_buckets = 6000

net.ipv4.route.gc\_timeout = 100

net.ipv4.tcp\_syn\_retries = 1

net.ipv4.tcp\_synack\_retries = 1

net.core.somaxconn = 65535

net.core.netdev\_max\_backlog = 262144

net.ipv4.tcp\_timestamps = 0

net.ipv4.tcp\_max\_orphans = 262144

### 测试场景

#### 单点 - 写内存 - 5 publisher - 1000 subscriber - Qos.2 - 60秒并发测试

#### 单点 - 写磁盘 - 5 publisher - 1000 subscriber - Qos.2 - 60秒并发测试

这里只针对新增的emitter和mosquitto做对比

#### 3结点集群 - 写磁盘 - 5 publisher - 1000 subscriber - Qos.2 - 60秒并发测试

Mosca本身没有cluster模式，这里不作比较，以下是emitter、emqtt、mosquitto的对比。