MQTT客户端库-Paho GO

2018-04-21 / VIEWS: 125

为了加深理解，本文是翻译文章。原文地址

Paho GO Client	
语言	GO
协议	EPL AND EDL
官网地址	http://www.eclipse.org/paho/
API类型	Asynchronous

描述

回到2013年10月，我转到了关于其他Paho MQTT客户端库的编写，并选择了作为一种新的语言去学习GO，还有什么比写一个MQTT客户端更好的方法去学习？该项目始于两个同事，病在2014年1月提交，并且作为开源项目持续更新的现在。

Paho GO 库还包含一个可以作为独立读写MQTT的包。

PAho Go 库目前是0.9版本，即将释放1.0的稳定版本，由于被商业和开源项目采用（例如Gobot ），该项目被积极的维护。

特性

MQTT3.1	✓	Qos 0	✓
MQTT3.1.1	✓	Qos 1	✓
LWT	✓	Q0s 2	✓
SSL/TLS	✓	Authentication	✓
Automatic Reconnect	✓	Throttling	✗

使用

安装

假设你有一个Go的开发环境，你有一个很简单的方法获取Paho Go库并运行；

```
go get git.eclipse.org/gitroot/paho/org.eclipse.paho.mqtt.golang.git
```

如下会将库下载到你的$GOPATH/src 目录下，你就可以在你的项目下添加到你的Import列表下使用该库：

```
git.eclipse.org/gitroot/paho/org.eclipse.paho.mqtt.golang.git
```

连接

```
opts := mqtt.NewClientOptions().AddBroker("tcp://broker
.hivemq.com:1883").SetClientID("sample")

c := mqtt.NewClient(opts)
if token := c.Connect(); token.Wait() && token.Error()
!= nil {
    panic(token.Error())
}
```

为了连接到MQTT代理，你必须提供两个必要的参数：代理的URL和使用的客户端ID。为此，我们创建了一个新的ClientOptions结构体实例，该结构体包含代理的Url和客户端ID。在ClientOptions结构体上操作的方法们返回一个可更改的结构体指针，这使你可以将方法连在一块。

参考了Paho Java 库，Paho Go 库允许你在完成操作时很容易的接受一个token，该token可以被用来指示操作是否完成。token.Wait()是个阻塞函数，只有在操作完成时才返回。token.WaitTimeout()会在操作完成后等待几毫秒后返回。

连接到MQTT3.1或MQTT3.1.1

```
opts := mqtt.NewClientOptions().AddBroker("tcp://broker
.hivemq.com:1883").SetClientID("sample")
opts.SetProtocolVersion(4)

c := mqtt.NewClient(opts)
if token := c.Connect(); token.Wait() && token.Error()
!= nil {
    panic(token.Error())
}
```

Paho Go 库默认使用MQTT3.1.1协议连接代理，如果失败他会自动回调并使用MQTT3.1协议连接。SetProtocolVersion()方法允许你明确的设置连接协议，4是3.1.1，3是3.1。如果显示设置，那么回调机制是禁用的。

使用LWT（临终遗嘱）连接

> *LWT:该协议提供了检测方式，利用KeepAlive机制在客户端异常断开时发现问题。因此当客户端电量耗尽、崩溃或者网络断开时，消息代理会采取相应措施。*
>
> *客户端会向任意点的消息代理发送“临终遗嘱”（LWT）信息，当消息代理检测到客户端离线（连接并未关闭），就会发送保存在特定主题上的 LWT 信息，让其它客户端知道该节点已经意外离线。*

```
opts := mqtt.NewClientOptions().AddBroker("tcp://broker
.hivemq.com:1883").SetClientID("sample")
opts.SetWill("my/will/topic", "Goodbye", 1, true)

c := mqtt.NewClient(opts)
if token := c.Connect(); token.Wait() && token.Error()
!= nil {
    panic(token.Error())
}
```

Paho Go库有两个方法设置LWT。SetWill()和SetBinaryWill(),这两个方法都有四个参数。两个方法中的第一个参数都是字符串型的LWT订阅。第二个参数是消息体(payload),在SetWill()中是一个字符创型，在SetBinaryWill()中是byte数组。第三个参数是消息的qos类型，第四个参数是LWT的是否保持连接布尔值。

使用用户名/密码连接

```
opts := mqtt.NewClientOptions().AddBroker("tcp://broker
.hivemq.com:1883").SetClientID("sample")
opts.SetUsername("username")
opts.SetPassword("password")

c := mqtt.NewClient(opts)
if token := c.Connect(); token.Wait() && token.Error()
!= nil {
    panic(token.Error())
}
```

发布

```
c.Publish("test/topic", 1, false, "Example Payload")

if token := c.Publish("test/topic", 1, false, "Example
Payload"); token.Wait() && token.Error() != nil {
    fmt.Println(token.Error())
}
```

上述中，c是mqtt.NewClient()返回的mqtt.Client。发布 使用4个参数；发布消息的字符串型的topic,消息的qos质量，是否保持消息连接的bool，或者既可以是字符串形式也可以是byte数组的消息体（payload）。并且我示范了如何使用和不适用token进行消息发布。

发布保留连接信息

```
c.Publish("test/topic", 1, true, "Example Payload")
```

订阅

```
var msgRcvd := func(client *mqtt.Client, message mqtt.M
```

```
essage) {
    fmt.Printf("Received message on topic: %s\nMessage:
 %s\n", message.Topic(), message.Payload())
}

if token := c.Subscribe("example/topic", 0, msgRcvd); t
oken.Wait() && token.Error() != nil {
    fmt.Println(token.Error())
}
```

Subscribe()使用3个参数，一个订阅的字符串形式的topic，订阅的qos质量和一个在接受到匹配订阅消息时的函数回调。回调函数必须有一个func(*mqtt.Client,mqtt.Message)的结构。当回调函数为空（nil）的时候，在库接受到消息后会调用客户端的默认消息处理进程（如果设置）。可以在结构体ClientOptions的SetDefaultPublishHandler()中设置。

取消订阅

```
c.Unsubscribe("example/topic")

if token := c.Unsubscribe("example/topic"); token.Wait(
) && token.Error() != nil {
    fmt.Println(token.Error())
}
```

Unsubscribe()可以接受多余一个的取消订阅的topic的参数，每个topic使用单独的字符串型数组参数分开。

断开连接

```
c.Disconnect(250)
```

Disconnect()使用一个参数，该参数为线程中结束任何工作的毫秒数。

使用SSL/TLS

```
opts := mqtt.NewClientOptions().AddBroker("ssl://iot.eclipse.org:8883").SetClientID("sample")

c := mqtt.NewClient(opts)
if token := c.Connect(); token.Wait() && token.Error() != nil {
    panic(token.Error())
}
```

你可以非常简单的通过改变代理URL来连接到具有SSL/TLS的代理；ssl，tls或者tcps都被client支持并且安全的连接。此处假设你连接的是使用系统已知证书的代理。如果你使用自我签名的证书你需要使用 TLS。使用ClientOptions的SetTlsConfig()配置。Paho Go库的Sample文档夹中有此示例代码。

关键词：token mqtt paho 使用 连接 error go opts 消息 topic

相关推荐：

Latest Paho Status (2)

Latest on the Eclipse Paho Project

原 荐 MQTT协议的初浅认识之推送订阅

mqtt介绍和go代码实现

Using MQTT V5 with the IBM Watson IoT Platform and the Eclipse Paho C client

IoT / SmartHome - Send EnOcean sensor data to Kafka

使用paho的MQTT时遇到的重连导致订阅无法收到问题和解决

Using Syslog with HiveMQ

12 Questions you should ask your broker vendor about MQTT

clustering

IOT: Should I use MQTT or HTTP?

Vevi-DP

Read more posts by this author.

China */source/cbb/veviDP*

Share this post