

消息队列 IoT MQ

快速入门

产品文档



腾讯云

【版权声明】

©2013-2018 腾讯云版权所有

本文档著作权归腾讯云单独所有，未经腾讯云事先书面许可，任何主体不得以任何形式复制、修改、抄袭、传播全部或部分本文档内容。

【商标声明】

及其它腾讯云服务相关的商标均为腾讯云计算（北京）有限责任公司及其关联公司所有。本文档涉及的第三方主体的商标，依法由权利人所有。

【服务声明】

本文档意在向客户介绍腾讯云全部或部分产品、服务的当时的整体概况，部分产品、服务的内容可能有所调整。您所购买的腾讯云产品、服务的种类、服务标准等应由您与腾讯云之间的商业合同约定，除非双方另有约定，否则，腾讯云对本文档内容不做任何明示或模式的承诺或保证。

文档目录

快速入门

使用入门

客户端签名计算

快速入门 使用入门

最近更新时间：2017-12-08 17:21:37

1. 查看 IoT MQ 实例

申请通过后，您的 IoT MQ 控制台中会展示 IoT MQ 实例，单击实例ID可以查看实例详情，包括接入点域名、实例规格等信息。

实例列表：

MQ for IoT 广州 上海 北京 MQ for IoT帮助文档							
请输入ID/名称 Q ↓							
ID	状态	接入点域名	类型	在线连接数	消息收发TPS	订阅关系数	消息生命周期
mqtt-5ox6fzsaa	运行中	mqtt-5ox6fzsaa.mqtt.myqcloud.com	微型	500	1,000	5,000	1天
mqtt-5p50b9y7s	运行中	mqtt-5p50b9y7s.mqtt.myqcloud.com	微型	500	1,000	5,000	1天

实例详情：

< mqtt-5ox6fzsaa	
基本信息	topic管理
基本信息	
ID	mqtt-5ox6fzsaa
接入点域名	mqtt-5ox6fzsaa.mqtt.myqcloud.com
地域	广州
规格	微型
同时在线连接数	500
消息收发TPS	1,000
订阅关系数	5,000
消息生命周期	1天
创建时间	2017-11-22 15:39:29
到期时间	2018-03-02 15:39:29

2. 管理 Topic

在 Topic 管理页面，您可以创建 Topic，指定 Topic 的名称、消息生命周期和消息最大长度。

注：当前 Topic 名称输入后无法更改。

新建 Topic :

消息生命周期

消息最大长度

新建Topic

×

名称

test/sensor/zone/001

英文大小写字母、数字、下划线 和正斜杠/,长度不超过128个字符

消息生命周期 ⓘ

1

天

范围 : 1天 - 7天

消息最大长度 ⓘ

64

K

范围 : 1K - 64K

提交

关闭

Topic 列表 :

mqtt-5ox6fzsaa

基本信息topic管理

新建

请输入名称

Q

↓

名称	消息生命周期	消息最大长度	操作
test/sensor/zone/001	1天	64K	编辑 删除
test/sensor/zone/002	1天	32K	编辑 删除
test/sensor/zone/003	1天	32K	编辑 删除

3. 生产消费

创建好 Topic 后，您可以通过 MQTT 客户端对发布订阅 Topic，进而进行生产和消费。

消息队列 IoT MQ 提供的 MQTT 服务严格遵循 MQTT3.1.1 协议，理论上能够适配所有 MQTT 客户端，但不排除客户端可能出现兼容性的问题。针对常用的用户场景，推荐的第三方包如下：

使用平台	第三方SDK	链接
Java	Eclipse Paho Java Client	http://www.eclipse.org/paho/clients/java/
Android	Eclipse Paho Android Service	http://www.eclipse.org/paho/clients/android/
JavaScript	Eclipse Paho JavaScript Client	http://www.eclipse.org/paho/clients/js/
Python	Eclipse Paho Python Client	http://www.eclipse.org/paho/clients/python/
GoLang	Eclipse Paho Go Client	http://www.eclipse.org/paho/clients/golang/
C++	MQTT C++ Client for Posix and Windows	http://www.eclipse.org/paho/clients/cpp/
.Net(C#)	C# .Net and WinRT Client	http://www.eclipse.org/paho/clients/dotnet/

注：

- 第三方 SDK 均可从 <http://www.eclipse.org/paho/downloads.php> 进行下载。
- 使用客户端连接 IoT MQ 服务器时，须发送 CONNECT 报文，且在 Connect 报文中需上传 username 和 password，其中 username 和 password 的计算详情参见[客户端签名计算](#)。

客户端签名计算

最近更新时间：2018-08-27 17:19:34

使用 MQTT 收发消息，服务端需要对客户端的身份进行权限校验，因此客户端请求中都需要带上签名以便比对身份。

腾讯云 IoT MQ 服务端会对每个客户端的访问请求进行身份验证，即客户端的每个请求中都需要包含签名信息（Signature），以验证用户身份。

1. MQTT SDK 访问服务器

MQTT 客户端连接 IoT MQ 服务器时，须发送 CONNECT 报文，且在 Connect 报文中需上传 username 和 password。其中 username 是 SecretId，password 是将 Appid、Instanceid 等作为待签名字符串，用 SecretKey 作为密钥计算得到的签名。

2. 申请安全凭证

在第一次使用客户端之前，用户需要在【腾讯云控制台】>【[API 密钥管理](#)】上申请安全凭证。安全凭证包括 SecretId 和 SecretKey，其中：

- **SecretId**：用于标识 API 调用者身份；
- **SecretKey**：用于加密签名字符串和服务器端验证签名字符串的密钥。

注意：

SecretKey 是构建腾讯云 MQTT 请求的重要凭证，使用腾讯云 MQTT 请求 可以操作您名下的消息队列 IoT MQ 资源，为了您的财产和服务安全，请妥善保存并定期更换密钥，当您更换密钥后，请及时删除旧密钥。

2.1 申请安全凭证步骤：

1. 登录 [腾讯云控制台](#)。

2. 单击【云产品】，选择【管理工具】栏下的【云 API 密钥】，进入云 API 密钥管理页面。



3. 在 [API 密钥管理](#) 页面，单击【新建密钥】即可以创建一对 SecretId/SecretKey。

注意：

- 开发商帐号最多可以拥有两对 SecretId / SecretKey。
- 被开发商添加为子用户的 QQ 帐号，在不同开发商控制台，可以申请不同的安全凭证。
- 子用户的安全凭证，目前仅可调用部分接口的云 API。

3. 生成签名串

有了安全凭证 SecretId 和 SecretKey 后，就可以生成签名串了。您可以选择在控制台上用工具生成签名，也可以使用签名算法自主计算签名。

3.1 使用控制台生成签名

为了方便用户对比验证自己的签名计算是否正确，IoT MQ 控制台提供了[客户端签名计算工具](#)供参考对比。

输入使用账号的 SecretId, SecretKey，选择需要访问的实例，即可得到客户端连接该实例需使用的用户名和密码。

码。

消息队列 IoT MQ

IoT MQ

客户端签名计算

客户端签名计算 广州 上海 北京

SecretId

AKIDz8krbsJ5yKBZQpn74WFkmLPx3gnPhESA

SecretKey

Gu5t9xGARNpq86cd98joQYCN3Cozk1qA

在 [云API密钥管理页面](#) 创建或找到已有SecretId/SecretKey

实例

mqtt-4wuymbpbs | mqtt-4wuymbpbs

计算结果

用户名 AKIDz8krbsJ5yKBZQpn74WFkmLPx3gnPhESA

密码 4SSm4Z8rVQZXDEMGAt5CFBA1rVTjYSPbs1lxqRJmnSs=

注：

此工具仅仅使用浏览器前端 JavaScript 完成计算，并不会传输 SecretKey 到 IoT MQ 后端，因此不用担心 SecretKey 泄漏的风险。

3.2 自主计算签名

拼接原串

假设用户的 Appid、Instanceid 分别是：

Appid : 1251762227

Instanceid : mqtt-4wuymbpbs

则拼接后的原串为：Appid=1251762227&Instanceid=mqtt-4wuymbpbs&Action=Connect

注意：

这里只是示例，请用户根据自己实际的 Appid 和 Instanceid 和请求参数进行后续操作。

使用签名算法生成签名串

假设用户的 SecretId、SecretKey 分别是：

SecretId：AKIDz8krbsJ5yKBZQpn74WFkmLPx3gnPhESA

SecretKey：Gu5t9xGARNpq86cd98joQYCN3Cozk1qA

注意：

这里只是示例，请用户根据自己实际的 SecretId 和 SecretKey 和请求参数进行后续操作。

然后用 SecretKey 作为密钥，使用 HmacSHA256 签名算法对上一步中获得的 签名原字符串 进行签名，然后将生成的签名串使用 Base64 进行编码，即可获得最终的签名串。

HmacSHA256 的算法实现，各个语言都有现成的函数库，下面以 PHP 语言为例（使用其它程序设计语言开发时，可用上述示例中的原字符串进行签名验证，得到的签名串与例子中的一致即可）：

```
$secretKey = 'Gu5t9xGARNpq86cd98joQYCN3Cozk1qA';  
$srcStr = 'Appid=1251762227&Instanceid=mqtt-4wuymbpbs&Action=Connect';  
$signStr = base64_encode(hash_hmac('sha256', $srcStr, $secretKey, true));  
echo $signStr;
```

最终得到的签名串为：

```
4SSm4Z8rVQZXDEMgAt5CFBA1rVTjYSPbs1lxqRJmnSs=
```

4. 客户端 CONNECT

使用客户端 SDK 连接 IoT MQ 服务器时，在 Connect 报文中的用户名为 SecretId，密码为签名串。根据上述示例，详情如下

username：AKIDz8krbsJ5yKBZQpn74WFkmLPx3gnPhESA

password：4SSm4Z8rVQZXDEMgAt5CFBA1rVTjYSPbs1lxqRJmnSs=