# 第7b章：采用MQTT 和亚马逊 Web 服务的云端连接

## 目标

学习第7b章后，你将掌握：

* 消息队列遥测传输（MQTT）的工作原理：
  + MQTT如何与TCP/IP相适配
  + **主题**（**Topic**）的作用

**消息代理（Message Broker**）的角色

* + **发布者**（**Publisher**）的角色
  + **订阅者**（**Subscriber**）的角色
* 如何使用JavaScript对象表示法（JSON）语言
* 亚马逊[AWS MQTT云服务的工作原理包括](https://docs.aws.amazon.com/iot/latest/developerguide/aws-iot-how-it-works.html)：
  + 如何通过创建物(*Things*)、策略(Policies)和证书(Certificates)在亚马逊AWS云服务中对物（从语义学的角度来看，本章中所说的*物*就是指物联网设备）进行定义；
  + AWS MQTT与和安全性；
  + 了解 *Thing* Shadow；
  + 如何使用AWS IoT MQTT测试客户端订阅和发布主题；
  + 理解能在AWS云端实施哪些系统（SNS、数据库等）。
* 详细了解如何编写可与AWS云交互的WICED固件

## 时间：4个小时

## 基础

### 亚马逊Web服务（AWS）

[AWS](https://aws.amazon.com/what-is-aws/)是一款高度安全的云服务平台，可提供计算能力、数据库存储、内容交付等服务（其为亚马逊带来的盈利高于零售业务）。AWS由众多组件精心打造而成，其中包含大量系列虚拟和实体的网络服务器，以及大量的Web服务器软件和管理工具，如（仅列出一部分）：

* AWS IoT：可为物联网设备提供云服务的云平台（本章的主题）；
* 亚马逊弹性云（Amazon Elastic Cloud）（[EC2](https://aws.amazon.com/ec2/)/[EC3](https://aws.amazon.com/ec3/)）：虚拟化的计算功能，基本全都是可租用的以Linux或Windows为主的服务器；
* [亚马逊Lambda](https://aws.amazon.com/lambda/)：可实现发送事件驱动型任务到云端执行的服务；
* 存储：大型快速的文件系统，包括[亚马逊 S3](https://aws.amazon.com/s3/?sc_channel=PS&sc_campaign=acquisition_US&sc_publisher=google&sc_medium=s3_b&sc_content=s3_e&sc_detail=amazon%20s3&sc_category=s3&sc_segment=105589469922&sc_matchtype=e&sc_country=US&s_kwcid=AL!4422!3!105589469922!e!!g!!a)和AWS弹性文件系统（[AWS Elastic File System](https://aws.amazon.com/efs/)）；
* 数据库：大型高速数据库，包括[Dynamo DB](https://aws.amazon.com/dynamodb/)、[亚马逊关系数据库（RDS](https://aws.amazon.com/rds/?sc_channel=PS&sc_campaign=acquisition_US&sc_publisher=google&sc_medium=rds_b&sc_content=rds_e&sc_detail=amazon%20rds&sc_category=rds&sc_segment=145495376109&sc_matchtype=e&sc_country=US&s_kwcid=AL!4422!3!145495376109!e!)）、[亚马逊 Aurora](https://aws.amazon.com/rds/aurora/?sc_channel=PS&sc_campaign=acquisition_US&sc_publisher=google&sc_medium=aurora_rds_b&sc_content=aurora_e&sc_detail=amazon%20aurora&sc_category=aurora&sc_segment=145509396189&sc_matchtype=e&sc_country=US&s_kwcid=AL)；
* 网络：高速、可容错、负载平衡的网络，接入点遍布全球；
* 开发工具：统一的编程接口(API)，可支持一系列不同的编程语言；
* 亚马逊简易通知系统[（SNS](https://aws.amazon.com/sns/)）：发送SMS和电子邮件等讯息的平台；
* [亚马逊简易队列服务（SQS](https://aws.amazon.com/sqs/)）：在服务器之间发送消息的平台（与MQTT消息不同）；
* [亚马逊 Kinesis](https://aws.amazon.com/kinesis/)：传输和分析“海量”数据的平台，是AWS物联网的数据管道。

### 亚马逊 AWS物联网云服务的简介

亚马逊AWS 物联网(IoT) 云服务包括MQTT消息代理和一系列服务器侧的功能性：

* 消息代理：MQTT消息代理；
* *物(thing)*的注册（*Thing* Registry）：可管理 *thing* 的存取的Web界面；
* 安全与身份：管理证书和*thing* 的规则的Web界面。用户能创建加密密钥并管理存取权限；
* “影子”(Shadow)：*thing* 的最新状态的在线缓存；
* 规则引擎：运行在云中的应用，能预订主题，并根据消息执行相应的编程操作。例如，经配置后可预订提醒(“Alert”)主题，如果某个物在这个提醒主题下发布了一条提醒消息，那么它就能通过亚马逊SNS向你的手机发送SMS文本消息；
* 物联网应用：具备可构建Web页面和手机App的SDK。

### 亚马逊AWS IoT云服务的资源

亚马逊AWS中有三种类型的资源：物 (*Thing*)、证书(Certificate)和策略(Policy)。第二讲将循序渐进地带你了解如何创建这三种类型。

**Thing**

*Thing* 代表某个设备或逻辑实体，其既能是一个物理设备或传感器（例如灯泡或墙上的开关），也可以是逻辑实体（如应用实例）或物理实体（不连接到AWS物联网云，但能与其它连接AWS物联网的设备有关联，如安装引擎传感器或控制面板的汽车）。

**Certificate**

AWS IoT云可在所有连接点实现相互验证与加密，这样就能确保数据必须经过身份验证才能在*物* 和AWS物联网之间进行交换。AWS物联网可支持基于X.509证书的验证。采用MQTT的连接也可使用基于证书的验证。用户可将策略（polices）连接至证书，从而允许或拒绝对AWS物联网云服务资源的访问。用户的设备可通过根证书授权（root CA）证书确保与实际的亚马逊 Web服务站点进行通信。你只能通过TLS将*物* 连接至AWS IoT云。

**Policy**

为连接网络的thing创建证书 (certificate)后，你必须创建并连接AWS IoT 策略（policy），从而决定thing可能执行什么样的AWS IoT 云操作。AWS IoT policy 为JSON文档，它们与AWS身份和存取管理policies遵循同样的惯例。

另外，你还能指定特定资源的许可，如topics与 shadow等。这里的策略实例是专为新的thing创建的。

{

"Version": "2012-10-17",

"Statement": [

{

"Action": [ "iot:\*" ],

"Resource": ["\*"],

"Effect": "Allow"

}

]

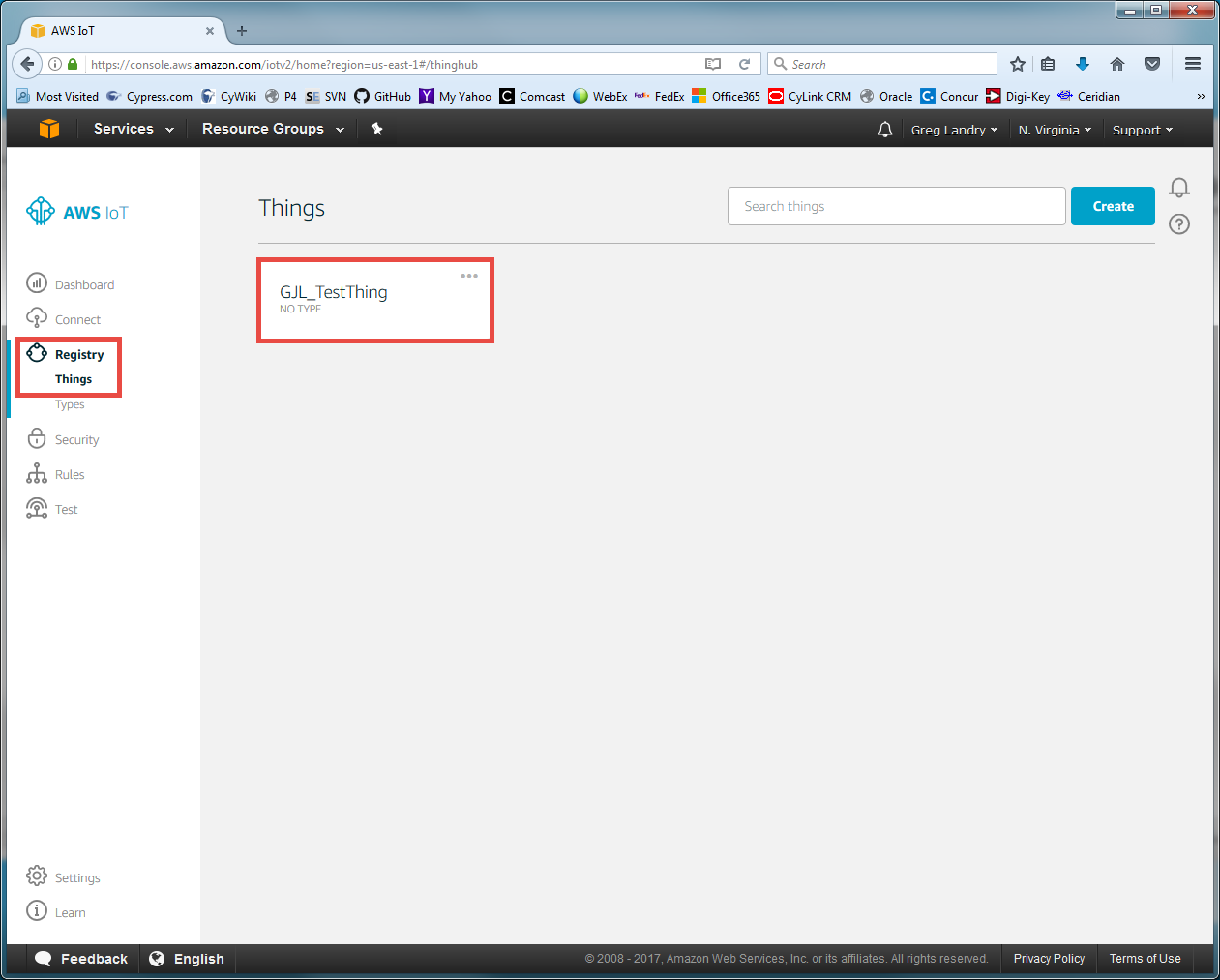
}

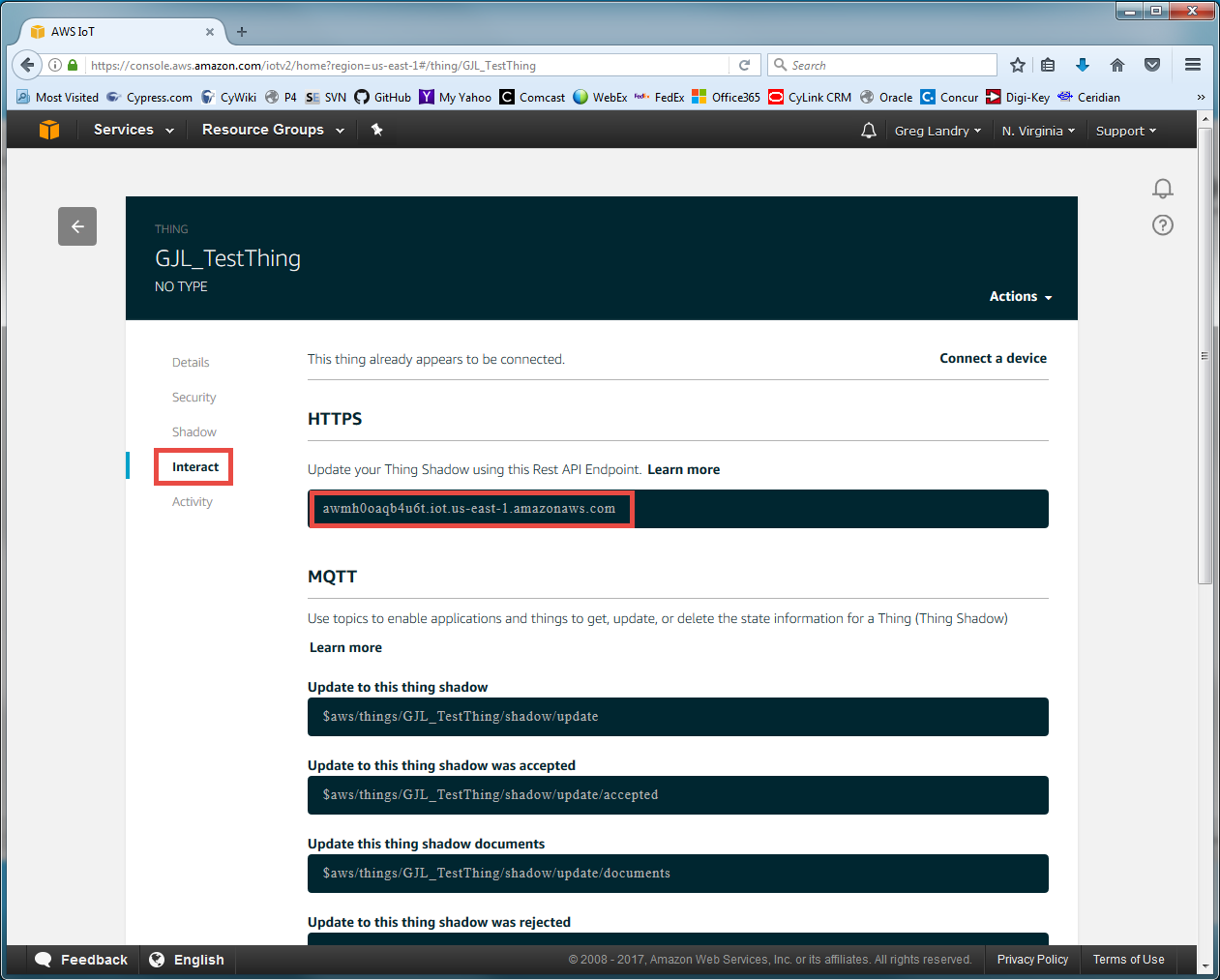
### 亚马逊 AWS MQTT

#### 互联网访问

在你创建AWS IoT 账户时，亚马逊将为你在云端创建一个新的（虚拟）服务器，并将在该服务器上打开MQTT消息代理。为了让你的WICED设备能接入该服务器，你将需要了解消息代理的IP地址。该地址将在固件中为MQTT\_BROKER\_ADDRESS。

为了查找地址，请从控制台选择你的Thing，然后再从左侧面板中选择交互（Interact）。地址将会列出为REST API终端。





[物的影子](http://docs.aws.amazon.com/iot/latest/developerguide/iot-thing-shadows.html) (*Thing* Shadow)

*Thing* Shadow有时也被表述为设备的影子 (Device Shadow)，就是一个JSON文档，可用于存储和检索*Thing*（设备、应用等）的当前状态信息。*Thing* Shadows服务可连接AWS IoT的每个*Thing*保留一个影子。你能采用*Thing*的影子通过MQTT或HTTP来获得并设置物的状态，无论当前*Thing*是否有互联网连接。每个*Thing* Shadow都可通过唯一的名称进行标识。

代表设备的JSON影子（JSON Shadow）文档具有以下属性：

* 状态 (State)：
  + 理想状态（desired）：*thing*的理想状态。应用能写入这部分文档，以便在无需直接与 *Thing*连接的情况下更新它的状态；
  + 报告状态 (reported)：*thing*的报告状态。*物*可写入这部分文档，以报告它们的最新状态。应用可读取这部分文档，以确定物的状态。
* 元数据 (Metadata)：在文档状态部分存储的数据信息，包括状态部分中每个属性的时间戳和Epoch时间，能帮助确定其更新的时间；
* 时间戳 (TimeStamp)：表明AWS物联网传输消息的时间。通过采用消息中的时间戳以及理想或报告部分的不同属性时间戳，一个*thing*能确定已更新项目的老旧程度，即便其并不具备内部时钟；
* 客户端令牌 (ClientToken)：设备唯一的字符串，能帮助你在MQTT环境中关联响应和请求；
* 版本：文档版本。每次更新文档时，版本号都会相应递增，这能确保文档版本的始终处于最新状态。

文档实例：



如果希望更新 Shadow，你可发布JSON文档，仅使用需要针对正确题目修改的信息，例如，你可以执行如下操作：

{

"reported": { "color": “BLUE” }

}

#### MQTT主题 (Topics)

AWS消息代理允许你创建几乎任何名称的主题，只有一个例外：命名为“$aws/…”的主题是由AWS IoT为特定功能预留的。

作为系统设计师，你需要定义topics在你的系统中的含义及功能，一些优良规范包括：

1. 不使用正斜杠；
2. 不使用空格；
3. 主题应当保持简明扼要；
4. 仅使用ASCII字符；
5. 嵌入唯一的标识符，如*Thin*g的名称等。

例如，适用于温度感应设备较好的主题名称可以是：myDevice/temperature。

[Device](https://docs.aws.amazon.com/iot/latest/developerguide/thing-shadow-mqtt.html) Shadow MQTT Topics

每个*thing*都拥有一组 topics，形式为“$aws/things/thingName/shadow/…”，有助于发布和订阅与shadow相关的topics。具体已经存在的 Shadow Topics包括：

|  |  |
| --- | --- |
| **MQTT Topic后缀** | **功能** |
| /update | 发布到这个topic的JSON消息将成为shadow的最新状态。 |
| /update/accepted | AWS将向这个主题发布一条消息，以作为/update的消息响应，说明已成功对影子进行了更新。 |
| /update/documents | 在文档通过/update发布更新后，将对这个主题发布完整的最新文档。 |
| /update/rejected | AWS将向这个主题发布一条消息，以作为/update消息的响应，说明影子拒绝更新。 |
| /update/delta | 将消息发送到/update后，如果理想状态与报告状态不同，那么AWS将发送一条JSON消息。该消息包含所有不匹配的属性。 |
| /get | 如果*物*向这个主题发布一条消息，那么AWS将向/get/accepted或/get/rejected发送消息作为响应，给出影子的当前状态。 |
| /get/accepted |
| /get/rejected |
| /delete | 如果*物*向这个主题发布一条消息，AWS将删除影子文档。 |
| /delete/accepted | 当/delete成功之后，AWS将向这个主题进行发布。 |
| /delete/reject | 当/delete拒绝后，AWS将向这个主题进行发布。 |

你可以使用“#”作为通配符来访问多个shadow topics。例如，可使用“$aws/things/thingName/shadow/#”为名称为“theThing”的*thing* 订阅所有的shadow topics。

## 练习题

### 00 在亚马逊IoT 控制台（console.aws.amazon.com）上运行MQTT教程

如果你没有控制台，那么你就需要注册亚马逊AWS账户。极简版账户可免费使用一年，但你必须提供信用卡信息才能开启账户。在一年免费期使用结束之前，请确保删除该账户，否则你将支付费用。

在Services菜单下，选择“AWS IoT”，点击“Get Started”，再点击“Start the tutorial”。





### 01 在AWS IoT 云中配置新的 thing，建立其policy和certificate，并使用AWS MQTT客户端进行测试

配置工作请遵循本章末尾的“AWS IoT 的*thing*配置流程”部分。

请注意消息代理的地址。当你点击你创建的 *thing*时，你就能在互动（Interact）页面看到它将被列为“REST API endpoint”。

测试消息代理请遵循本章末尾的“使用AWS MQTT客户端的流程”部分。

### 02 构建并测试demo.aws\_iot.pub\_sub.publisher App

1. 复制WICED apps/demo/aws\_iot/pub\_sub/publisher项目到你自己的目录（例如ww101/07b/02\_publisher）并更新所有文件。
   1. 提示：确保将你的平台添加到生成文件中的有效平台，并移除所有其他平台。
2. 为你的网络修改DCT。
3. 将你在01题目中生成的证书复制到resources/apps/aws\_iot目录。如下是对这些文件的重命名：
   1. 文件rootca.cer是你的应用将用来对发送给亚马逊的数据进行加密的公共密钥。这是AWS已知的良好密钥，内置于SDK中。由于它不会改变，因而无需下载。
   2. 文件client.cer是你的 *thing*的认证。AWS据此了解，这是试图进行通信的 *thing*。该认证在你设置*thing*的时候由AWS创建，文件名称为<name>-certificate.pem.crt。因此，你必须对你下载到client.cer的证书进行重命名，才能取代现有的client.cer文件。
   3. 文件privkey.cer是专用密钥，你的应用将用于解密从AWS发回的数据。在建立你的*thing* 时，它由AWS创建，文件名称为<name>-private.pem.key。因此，你必须对下载到privkey.cer的专用密钥进行重命名，才能取代现有的privkey.cer文件。

d. 由AWS创建的另一个文件命名为 <name>-public.pem.key。这是公用密钥，可在从AWS获取传回数据时匹配你的应用使用的私人密钥。你的应用无需这个密钥，因为它是当AWS给你发送的数据时加密用的。由于其最初由AWS生成，因而AWS已经知道密钥。

提示：如果你在构建项目后会更改密钥或证书，那么你必须在重构之前运行“Clean”(清除)命令，否则你的项目将得不到新的密钥。

1. 创建项目的make target (生成目标)。
2. 更改WICED\_TOPIC和MQTT\_BROKER\_ADDRESS的 #defines, 采用01题目里面的topic和 broker地址。
3. 创建项目并烧录你的项目。
4. 打开串行端口，并查看终端会话。
5. 使用AWS MQTT测试客户端订阅主题。当按下按键时，你应能看到主题的更新。

### 03 通过回答下列问题来详细阐述发布者app的固件流程：

1. MQTT库函数（如*wiced\_mqtt\_publish()*）如何导入项目？
2. 按下按键时调用什么函数？
3. 按键回调(Callback)函数是如何解锁主线程的？
4. “wait\_for\_response”采用何种WICED SDK RTOS机制进行“wait”？
5. 为什么固件作者创建了名为“wait\_for\_response”的函数？
6. 是否所有发送给AWS IOT MQTT消息代理的消息都为JSON格式？
7. 7个WICED MQTT事件是什么？它们是在哪个文件中进行定义的？
8. 客户端证书是否必须命名为client.cer？如何更改名称？
9. WICED MQTT库函数与publisher app中库函数的外围函数的之间，是使用何种命名规则进行区别？
10. 建立MQTT连接需要哪些步骤？
11. 什么能避免publisher app锁死造成连接断开？
12. 避免固件在发布完成前发送多次按键动作的旗标(flag)的名称是什么？

### 04构建并测试demo.aws\_iot.pub\_sub.subscriber App

1. 可从apps/demo/aws\_iot/pub\_sub/subscriber中将WICED应用复制到你的目录（如wa101/07b/05\_subscriber），并修改DCT和 makefile；
2. 更新 topic 和 broker #defines，使其与你在03题中选择的相同；
3. 我们将使用与02题中相同的 *thing*，因此证书和密钥也将会相同；
4. 使用AWS MQTT测试客户端发布消息。
   1. 确定开关灯需要发送什么字符串；
      1. 提示：查看源代码，以查找接收消息时寻找的字符串；
      2. 提示：如果你成功，那么基板上的绿光LED应打开或关闭。

### 05（高级）可用两个不同的套件实施订阅和发布，并进行测试

1. 在实际应用中，你通常会遇到一个或多个设备将数据发布到一个代理以及一个或多个设备从同一个代理读取数据的情况。因此，我们尝试使用两个不同的套件。你应与另一名学员形成一个小组，一起来做这个实验；
   1. 在AWS控制台为订阅者创建新的 *thing*，以确保订阅者和发布者的标识不同；
   2. 保存新的订阅者证书文件，但可为*client.cer*和*privkey.cer*设置不同的名称，以确保订阅者和发布者可被识别为两种不同的 *thing*；
   3. 更新生成文件，以确保订阅者可指向新的资格认证；
      1. 提示：无需更改rootca.cer，因为它是亚马逊AWS的公开密钥，应始终保持相同；
   4. 在“subscriber.c”中更新指向新的证书和密钥文件的行。
      1. 提示：可将证书列示为*resources\_apps\_DIR\_aws\_iot\_DIR\_client\_cer* 和*resources\_apps\_DIR\_aws\_iot\_DIR\_privkey\_cer*。上述名称是资源文件夹中的路径，其中的文件夹名称可用密码“\_DIR\_”分开，而在cer之前可用“\_”替代。你可将证书移动到资源文件夹中的其他位置， 或者遵循命名规则更改文件名称并将其放置在相同文件夹中。
   5. 对已更新的订阅者固件进行编程；
   6. 上电启动两个套件；
   7. 使用AWS MQTT测试客户端订阅你选择的主题；
   8. 按下供应方上面的按钮，观察订阅方上LED的状态变化情况。此外，还可观察MQTT客户端窗口中的消息。

### 06（高级）构建并测试 Shadow App

1. 本例采用的配置AP提供Web服务器，由此*thing*的名称、资格证书、密钥和网络设置能在连接到配置AP的一台设备上由Web浏览器完成；
2. 从*apps/demo/aws\_iot/shadow*中将WICED应用复制到你的目录中，并更新 makefile；
3. 更新DCT 中的Config AP提供一个唯一的SSID名称（不会与班级中其他人的发生冲突）名称；
4. 更新消息代理（message broker）地址，以匹配你在此前练习中创建的地址；
   1. 提示：消息代理地址在本项目中为“*aws\_common.h”*。
5. 烧录到开发板；
6. 把你的电脑通过Wi-Fi连接到你的开发板上的Config AP：
   1. 连接到开发板配置出的SSID；
   2. 进入Web服务器（当启动设备并开始AP时，将IP地址印在终端上）；
      1. 提示：不要使用火狐浏览器，它有时会给出奇怪的结果；
   3. 更新*thing*的名称，以匹配你在此前练习中创建的名称；
      1. 提示：显示的默认名称在“*aws\_config.h”*中，因此你也能在对开发板进行编程之前对其进行更改；
   4. 根据说明上传客户端证书和私有密钥；
   5. 点击“Wi-Fi Setup >”，并连接到Wi-Fi网络；
   6. 电路板将重启。完成重启后，电路板将作为一个站连接到你在此前练习中配置的Wi-Fi网络；
7. 从你的计算机连接至一个Wi-Fi接入点；
8. 连接到console.aws.amazon.com，并启动MQTT测试客户端；
9. 订阅设备的 shadow topics。
   1. 提示：*$aws/things/<YourThingName>/shadow/#* 将为你的*thing*订阅所有的影子主题。这里的 # 是一个通配符。
10. 按下电路板上的按键，并查看消息；
11. 问题：将LED从打开（On）变为关闭（Off）的事件时序是怎样的？

## 参考文献

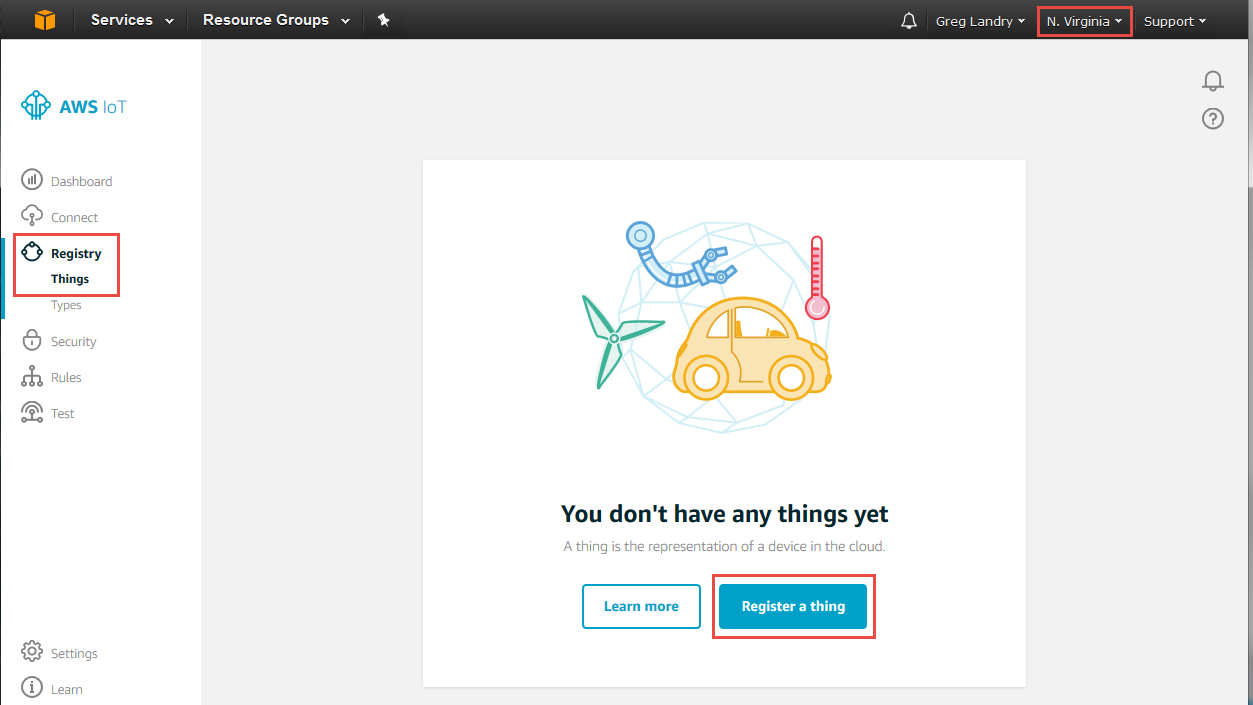
|  |  |
| --- | --- |
| **资源来源** | **链接** |
| [AWS](http://docs.aws.amazon.com/iot/latest/developerguide/what-is-aws-iot.html) Developers Guide | <http://docs.aws.amazon.com/iot/latest/developerguide/what-is-aws-iot.html> |
| [AWS IOT Getting Started](https://aws.amazon.com/iot/getting-started/) | https://aws.amazon.com/iot/getting-started/ |
| [A nice powerpoint about MQTT](http://www.slideshare.net/PeterREgli/mq-telemetry-transport) | http://www.slideshare.net/PeterREgli/mq-telemetry-transport |
| [MQTT Topic Naming Best Practices](http://www.hivemq.com/blog/mqtt-essentials-part-5-mqtt-topics-best-practices) | http://www.hivemq.com/blog/mqtt-essentials-part-5-mqtt-topics-best-practices |
| [Avnet Getting Started](http://cloudconnectkits.org/system/files/GSG-BCM4343W%20IoT%20Starter%20Kit%20-%20Getting%20Started%20%28v1.1%29.pdf) | http://cloudconnectkits.org/system/files/GSG-BCM4343W%20IoT%20Starter%20Kit%20-%20Getting%20Started%20%28v1.1%29.pdf |
| Avnet User Guide [Part1](http://cloudconnectkits.org/system/files/Tutorial%20Part1%20-%20Tool%20Install%2C%20USB%20drivers%20and%20AWS%20Shadow%20%28v1.1%29.pdf) and [Part2](http://cloudconnectkits.org/system/files/Tutorial%20Part2-App%20Development%20using%20WICED%20SDK-v1.1.pdf) |  |
| [AWS Forum](https://forums.aws.amazon.com/forum.jspa?forumID=210) | https://forums.aws.amazon.com/forum.jspa?forumID=210 |
|  |  |
|  |  |
|  |  |
|  |  |

## 相关实例“Apps”

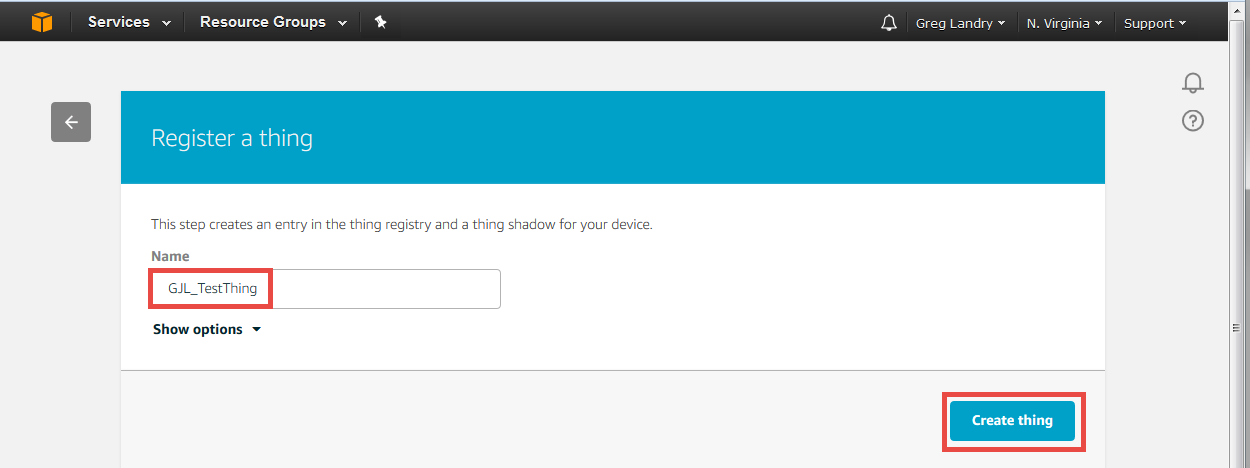
|  |  |
| --- | --- |
| **App名称** | **功能** |
| demo.aws\_iot\_pub\_sub/publisher | 向AWS云演示发布信息 |
| demo.aws\_iot\_pub\_sub/subscriber | 在AWS云中演示主题订阅 |
| demo.aws\_iot\_shadow | 演示如何使用AWS的shadow设备 |

## 配置AWS IoT *thing*的流程

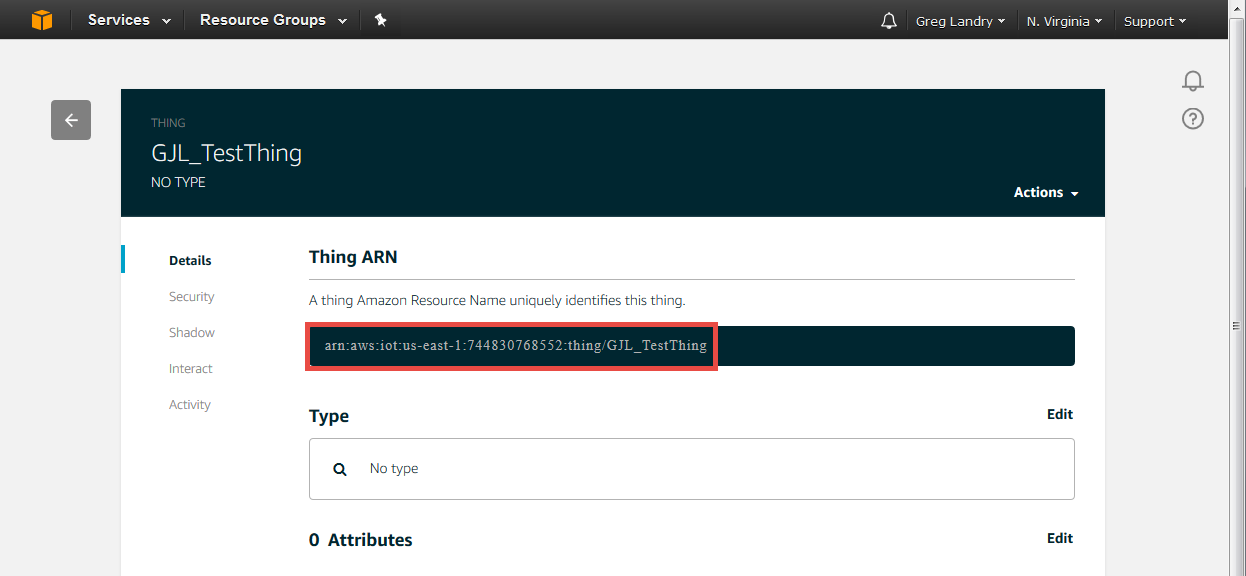
1. 观看console.aws.amazon.com的教程后，你应进入“Things”页面，点击“Register a thing”。
   1. 提示：当前只有美东时区才能正常工作，因此在设置新的*thing*之前请选择美东时区内的位置。在以下实例中，我们选择的是 N. Virginia。



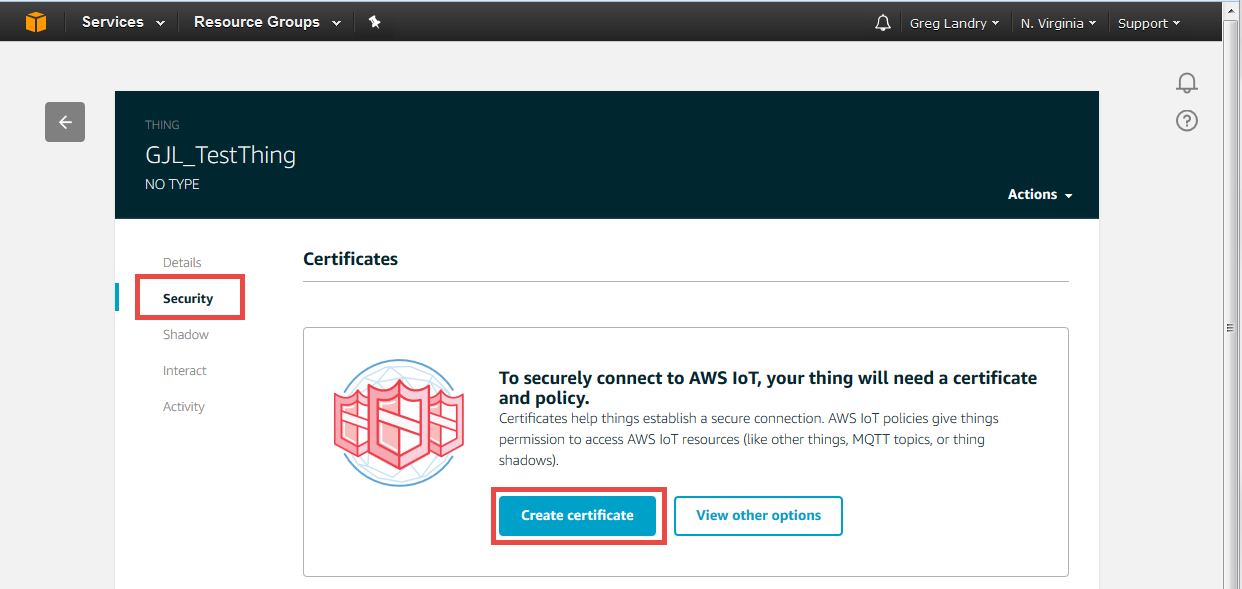
1. 将其命名为“<YourInitials>TestThing”（或其他任何名称）并按下“Create”。



1. 你现在应看到具有你的 *thing*的信息的窗口。复制你的 *thing*的完整的亚马逊资源名称（Amazon Resource Name）并保存，因为你随后还会需要它。

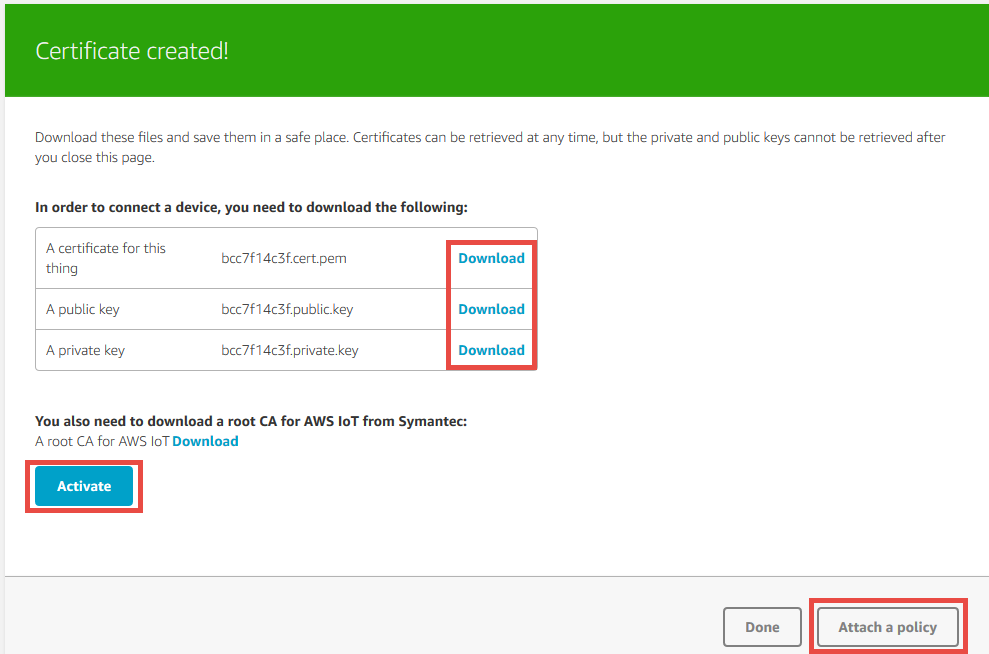


1. 在对 *thing* 进行任何操作前，你需要创建密钥，以帮助你识别作为*thing* 的身份，然后再更新数据。为了执行此操作，点击“Security”，随后再点击“Create Certificate”。



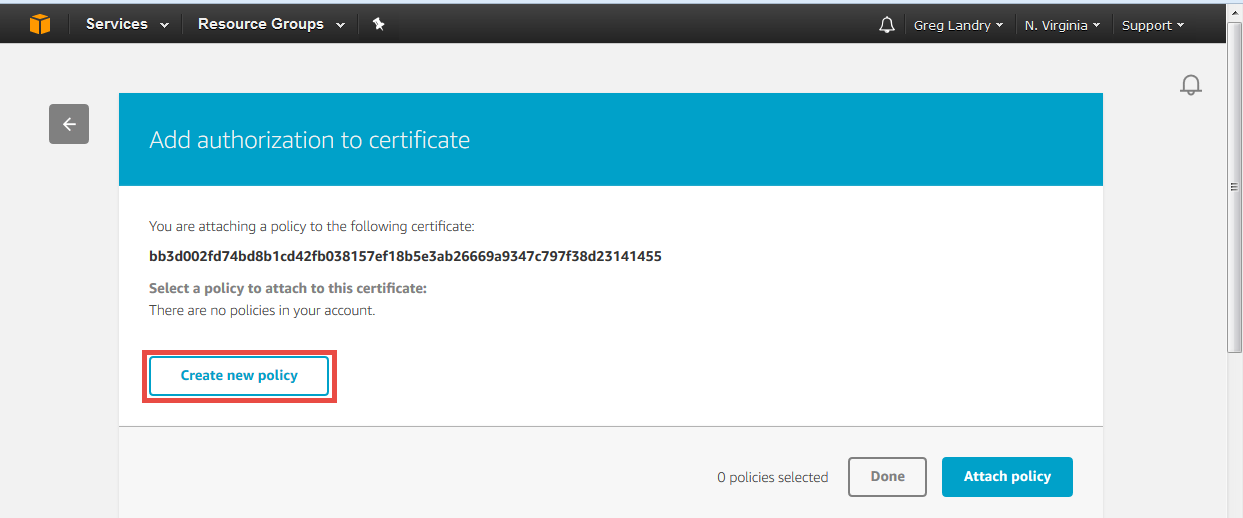
1. **现在你需要下载“证书”、“公共密钥”和“私有密钥”。如果你忘记这个步骤，你将不能返回……因此你必须现在就下载那些文件，才能确保TLS正常工作！此外，你还必须激活（Activate）证书。在下载密钥并激活证书后，就可点击“Attach a policy”。**

注释：该窗口还会给出可从Symantec（一个受信任的认证权威机构）为AWS物联网下载根CA的选项。但是，这一步并不是必需的，因为AWS物联网的根CA已经包含在WICED SDK中了。



**Do this first!**

1. 点击“Create new policy”。



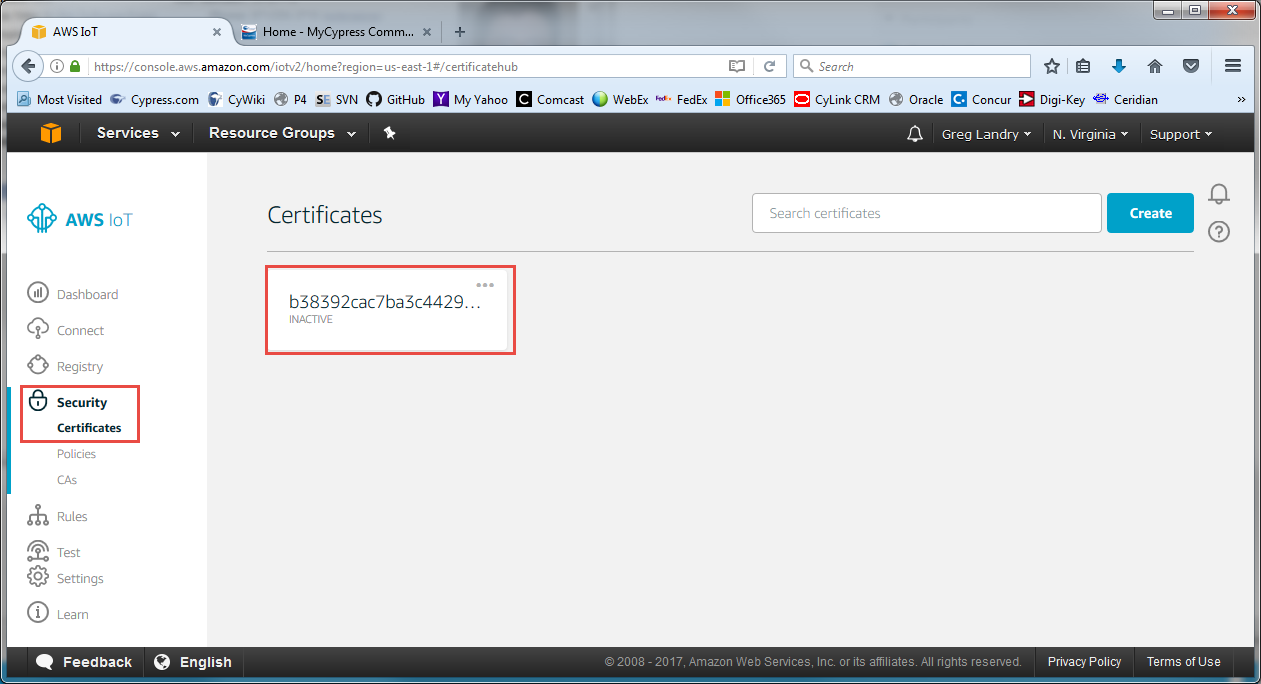
1. 为新的policy命名，如“<YourInitials>\_TestThing\_Policy”。添加可作为“*iot:\*”*的行为，为资源ARN（Resource ARN）输入“*\*”*，并选择“Allow”。随后再点击“Create”按键。



1. 你将看到policy文档的详细信息。在本例中，允许执行任何iot操作（iot:\*）。



1. 你现在需要将policy连接至certificate。首先点击屏幕左侧的左箭头，如上所示。随后从左侧面板中选择Security -> Certificates，再点击你的 certificate。



1. 看到你的 certificate后，选择“Actions -> Attach Policy”。选择你的策略，并点击“Attach”。完成这些操作后，点击左箭头。





## 使用AWS Test MQTT Client的流程

AWS Test MQTT Client是一个基于 Web浏览器的测试客户端，你能连接到“你的”消息代理。随后你就能发布和订阅主题。你可将其视为物联网上可以发布和订阅的 *thing*。使用它来测试的步骤如下：

1. 从屏幕左侧面板上选择“Test”。输入你想订阅的topic，如“testtopic”，并点击“Subscribe to topic”。你将在Subscriptions下方看到新的topic。



1. 既然已经订阅了topic，就能向该topic发布消息。为了进行此操作，请填写topic的名称“testtopic”，然后再输入你的消息，并按下“Publish to topic”。在以下框中可以看到，发送的“test message”。



1. 发送完消息后，你将在订阅（Subscriptions）区域中的主题附近看到个红点。这说明你在该订阅区域有一条新消息。点击testtopic以查看这条新消息。



1. 连接你的物联网设备后，测试客户端对测试订阅和发布操作非常实用，正如我们将在练习中看到的一样。