

# Cloud Tutorial: AWS IoT

---

CSE 521S Fall,  
Aug. 29, 2019  
Ruixuan Dai



Washington University in St. Louis

# Pointers

---

- Amazon IoT
  - <http://docs.aws.amazon.com/iot/latest/developerguide/what-is-aws-iot.html>
- Raspberry Pi
  - <https://www.raspberrypi.org/>
- Resource list for course projects
  - [http://cps.cse.wustl.edu/index.php>List\\_of\\_Projects](http://cps.cse.wustl.edu/index.php>List_of_Projects)
- Apply for \$40 credits for Amazon AWS
  - <https://aws.amazon.com/education/awseducate/apply/>

# XaaS: Basics in Cloud Computing



Washington University in St. Louis

# Cloud Computing

- Cloud computing provides **shared pool of configurable computing resource** to end users on demand
- Three service models
  - **IaaS (Infrastructure as a Service)**: virtual machines, storage, network ...
 

  - **PaaS (Platform as a Service)**: execution runtime, middleware, web server, database, development tool ...
 

  - **SaaS (Software as a Service)**: email, virtual desktop, games ...
 


# Cloud Services: On-premise Software

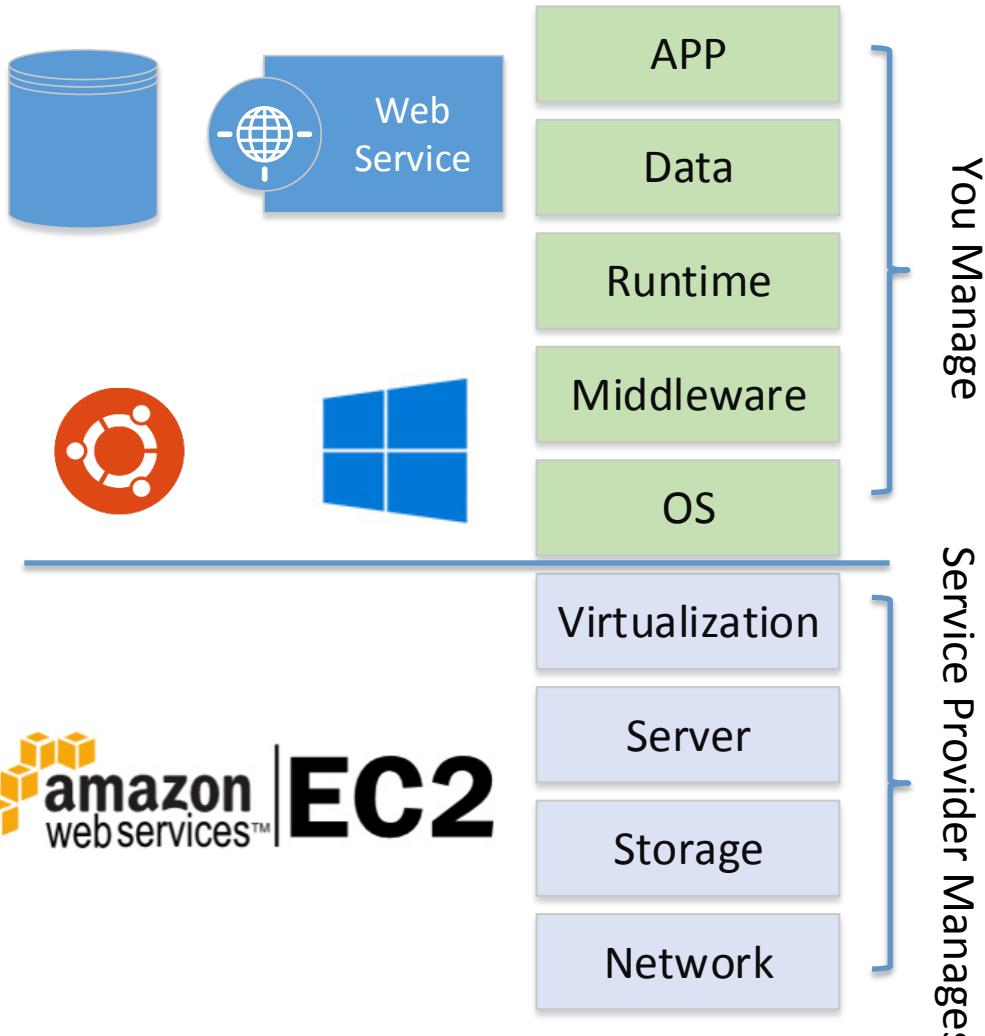
- Traditional
- installed and runs on personal computer
- You Manage and Deploy
  - ❑ Hardware
  - ❑ OS
  - ❑ Software
- Example
  - ❑ This presentation



# Infrastructure as a Service (IaaS)

## ➤ IaaS

- ❑ "physical server box"
- ❑ Virtual Machine
  - Memory
  - Storage
  - CPU
  - Network



## ➤ Example

- ❑ AWS EC2
- ❑ AWS EFS

## ➤ Use case

- ❑ Build up your VM cluster

# Platform as a Service (PaaS)

## ➤ PaaS

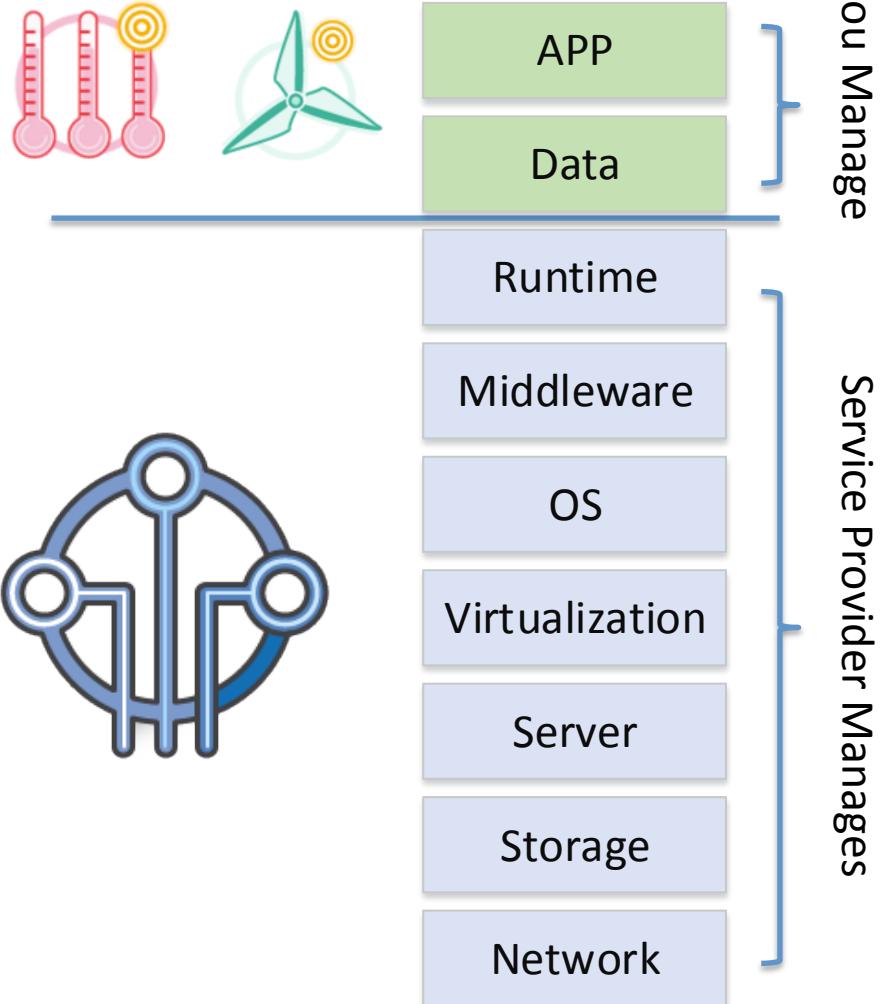
- ❑ You get a framework
- ❑ Host Application
- ❑ Tools

## ➤ Example

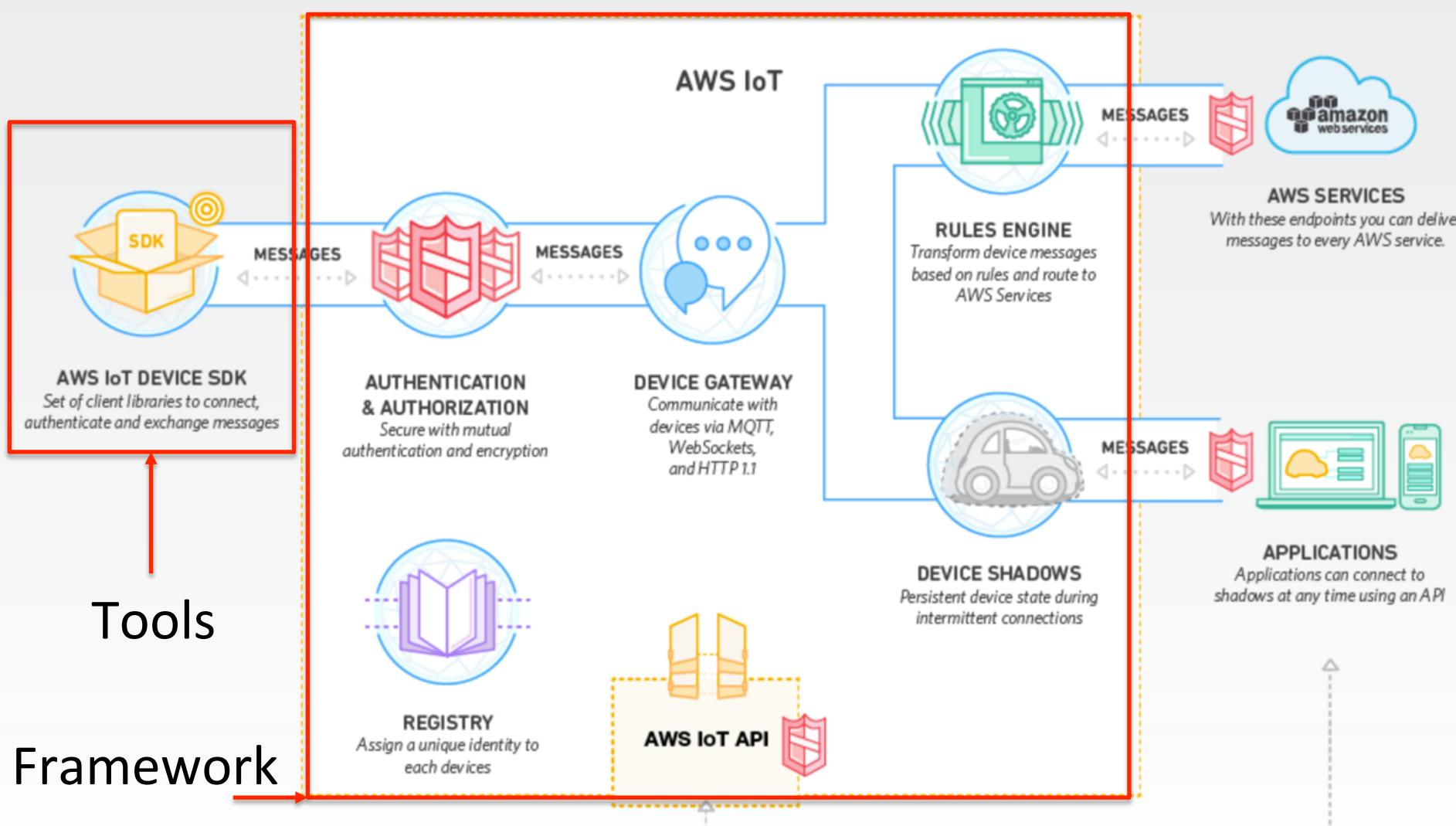
- ❑ AWS IoT

## ➤ Use case

- ❑ Build up your smart A/C controller



# PaaS Example: Amazon IoT



# Software as a Service (SaaS)

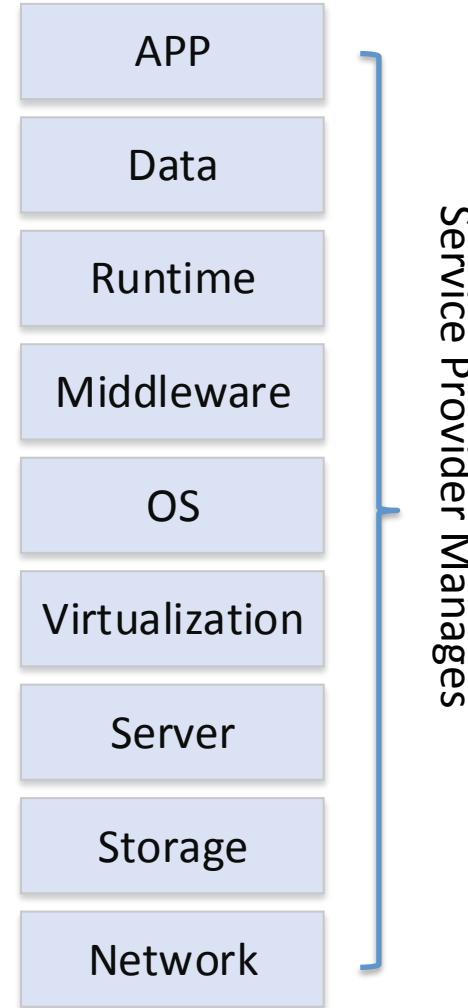
## ➤ SaaS

- ❑ You get a whole solution

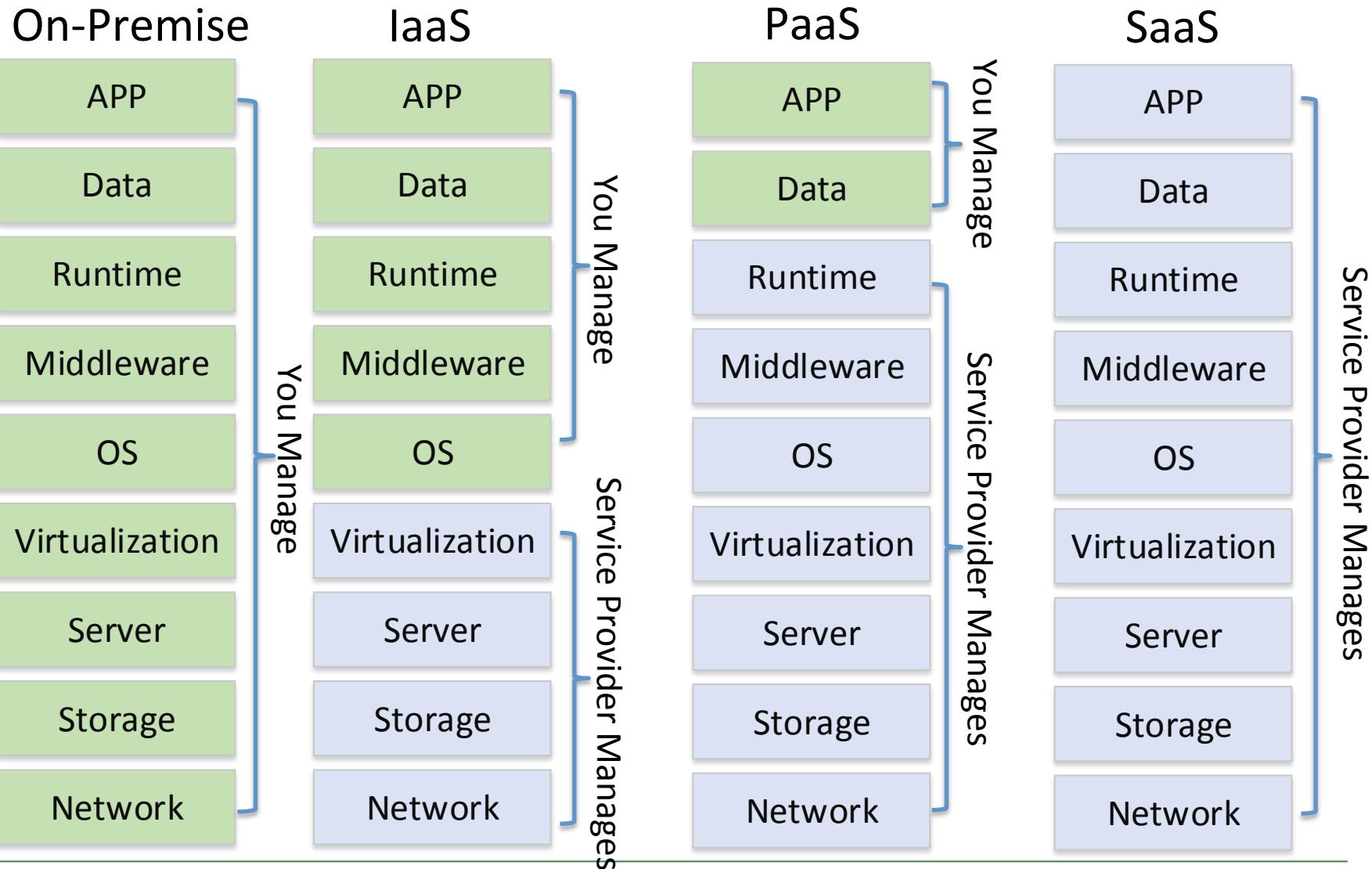


## ➤ Example

- ❑ Gmail
- ❑ Dropbox
- ❑ Office365
- ❑ Stadia



# XaaS: A Recap



# Tutorial: Hello! AWS IoT!!

---



Washington University in St. Louis

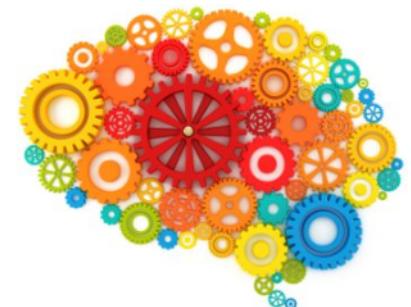
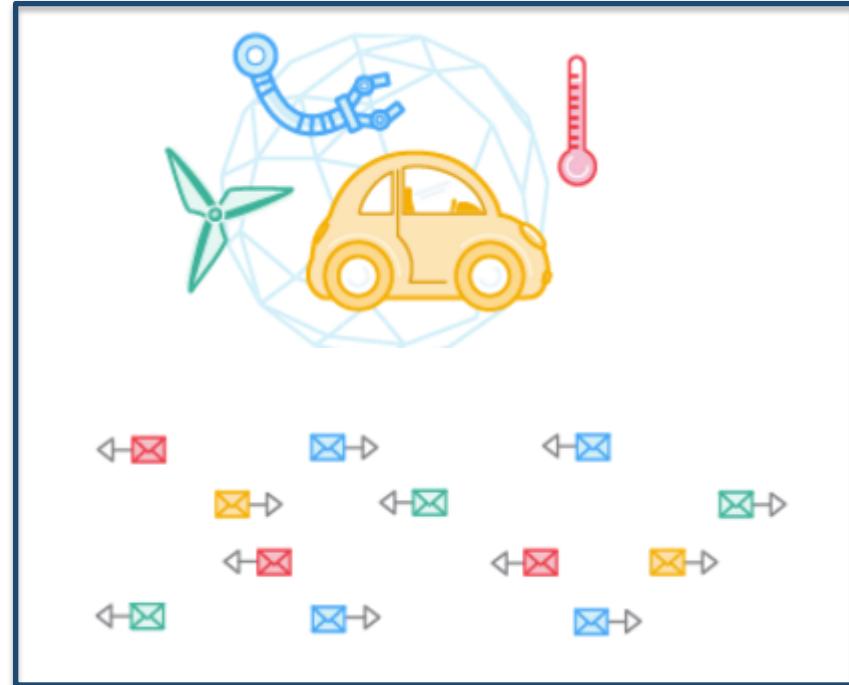
# Internet-of-Things

## ➤ Things (Devices)

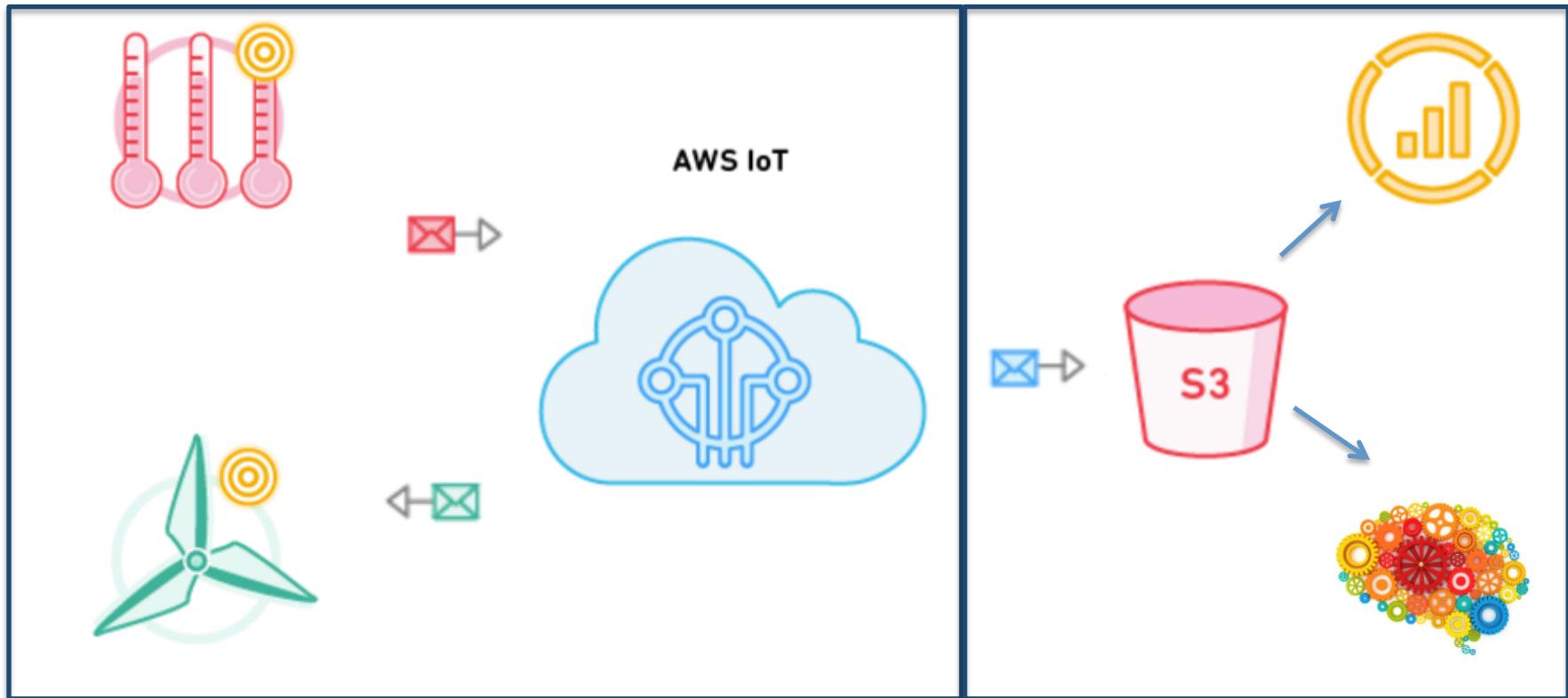
- Many of them
  - Different Types
  - Isolated Systems

- Data and Command
  - Sensing the world
  - Give Response

- Challenge
  - United: Connected + Communication
  - Smart: Data Analytics + Strategy



# Solution: AWS IoT

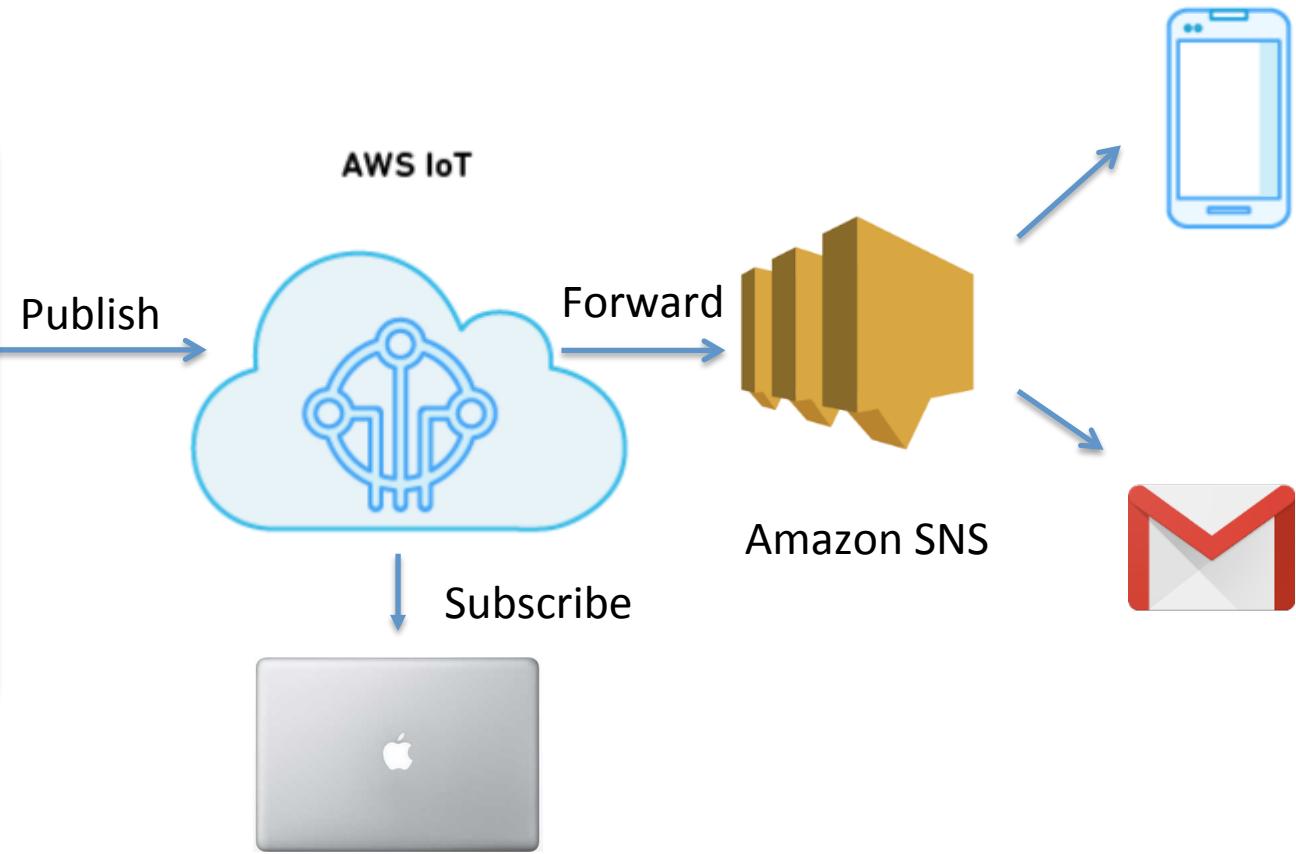
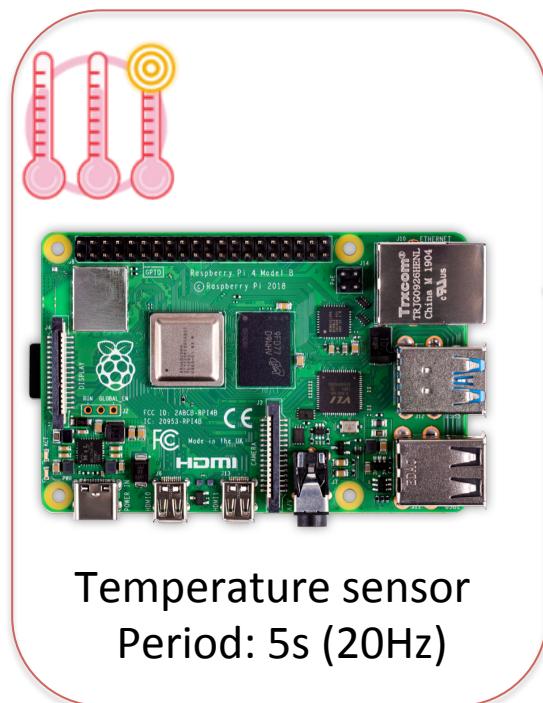


United: Connect + Communication

Stated: “Thing Shadow”

Smart: Other Cloud Service  
Data Storage  
Machine Learning

# Tutorial: Hello AWS IoT!



# Get into AWS Manage Console

- Create your own AWS account
- Sign In IoT Manage Console
  - <https://aws.amazon.com/iot/>

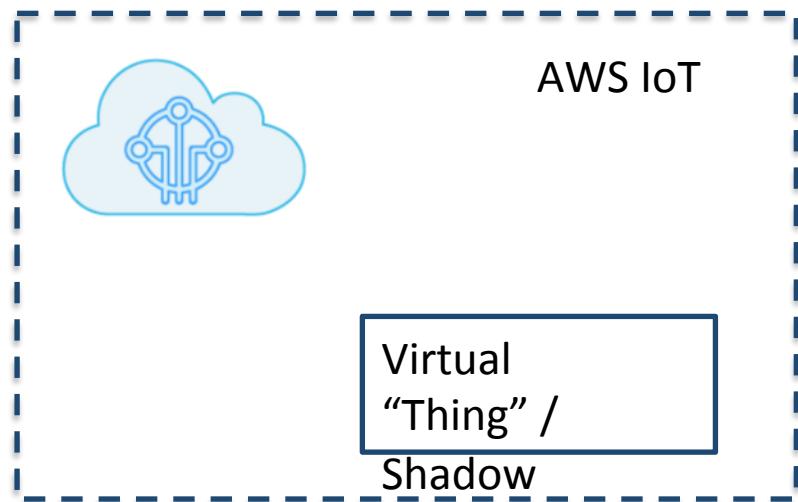
Group
A-Z

 <b>Compute</b> <ul style="list-style-type: none"> <li>EC2</li> <li>Lightsail ↗</li> <li>ECR</li> <li>ECS</li> <li>EKS</li> <li>Lambda</li> <li>Batch</li> <li>Elastic Beanstalk</li> <li>Serverless Application Repository</li> </ul>	 <b>Robotics</b> <ul style="list-style-type: none"> <li>AWS RoboMaker</li> </ul>	 <b>Analytics</b> <ul style="list-style-type: none"> <li>Athena</li> <li>EMR</li> <li>CloudSearch</li> <li>Elasticsearch Service</li> <li>Kinesis</li> <li>QuickSight ↗</li> <li>Data Pipeline</li> <li>AWS Glue</li> <li>AWS Lake Formation</li> <li>MSK</li> </ul>	 <b>Business Applications</b> <ul style="list-style-type: none"> <li>Alexa for Business</li> <li>Amazon Chime ↗</li> <li>WorkMail</li> </ul>
 <b>Blockchain</b> <ul style="list-style-type: none"> <li>Amazon Managed Blockchain</li> </ul>		 <b>End User Computing</b> <ul style="list-style-type: none"> <li>WorkSpaces</li> <li>AppStream 2.0</li> <li>WorkDocs</li> <li>WorkLink</li> </ul>	
 <b>Storage</b> <ul style="list-style-type: none"> <li>S3</li> <li>EFS</li> <li>FSx</li> </ul>	 <b>Management &amp; Governance</b> <ul style="list-style-type: none"> <li>AWS Organizations</li> <li>CloudWatch</li> </ul>	 <b>Security, Identity, &amp; Compliance</b> <ul style="list-style-type: none"> <li>IAM</li> <li>Resource Access Manager</li> </ul>	 <b>Internet Of Things</b> <ul style="list-style-type: none"> <li>IoT Core</li> <li>Amazon FreeRTOS</li> <li>IoT 1 Click</li> </ul>

# Step 1: Create a Virtual "Thing"

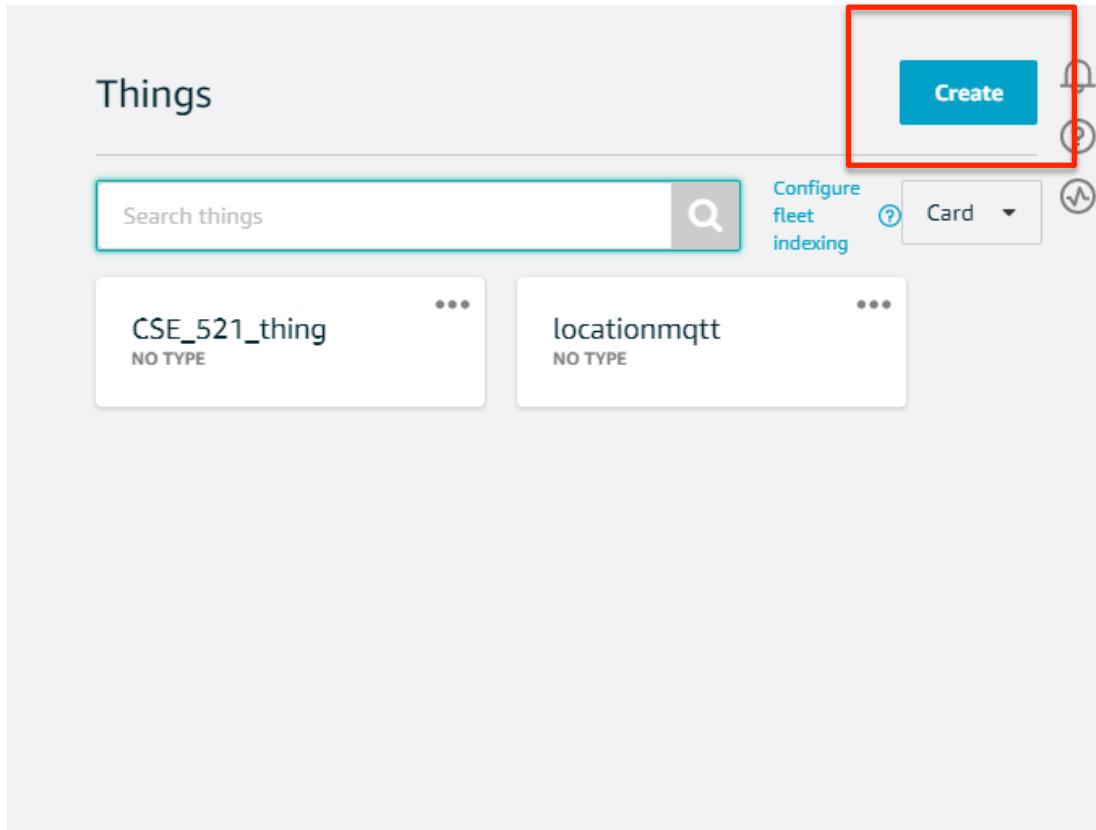
- A Thing in AWS IoT has a “**shadow**”
  - a JSON document that is used to **store and retrieve current state information for a device.**
    - **E.g.** Battery level, Connectivity, data

A “Dashboard” to show some info
  - **Shadow is a special topic in AWS IoT**
- **Certificates and policy**
  - **Authentication, Security**
  - **Permission and roles**



# Create a thing

- 1. AWS IoT Menu
  - Things → Create
- 2. Give a name



The screenshot shows the AWS IoT Things interface. On the left, there's a sidebar with the AWS IoT logo and several menu items: Monitor, Onboard, Manage (with sub-options Things and Types), Thing Groups, Billing Groups, Jobs, Greengrass, Secure, Defend, and Act. The 'Manage' section is highlighted with a red box. In the main area, there's a 'Things' heading, a search bar, and two things listed: 'CSE\_521\_thing' and 'locationmqtt'. A large red box highlights the 'Create' button in the top right corner of the main panel.

This step creates an entry in the thing registry and a thing shadow for your device.

## Name

## Apply a type to this thing

Using a thing type simplifies device management by providing consistent registry data for things that share a type. Types provide things with a common set of attributes, which describe the identity and capabilities of your device, and a description.

### Thing Type

## Add this thing to a group

Adding your thing to a group allows you to manage devices remotely using jobs.

### Thing Group

## Set searchable thing attributes (optional)

Enter a value for one or more of these attributes so that you can search for your things in the registry.

### Attribute key

### Value

## Show thing shadow ▾

CREATE A THING

## Add a certificate for your thing

STEP  
2/3

A certificate is used to authenticate your device's connection to AWS IoT.

### One-click certificate creation (recommended)

This will generate a certificate, public key, and private key using AWS IoT's certificate authority.

[Create certificate](#)

### Create with CSR

Upload your own certificate signing request (CSR) based on a private key you own.

[!\[\]\(eca9dca13a2688a2d5e75f35d4cc16aa\_img.jpg\) Create with CSR](#)

### Use my certificate

Register your CA certificate and use your own certificates for one or many devices.

[Get started](#)

### Skip certificate and create thing

You will need to add a certificate to your thing later before your device can connect to AWS IoT.

[Create thing without certificate](#)

## Certificate created!

Download these files and save them in a safe place. Certificates can be retrieved at any time, but the private and public keys cannot be retrieved after you close this page.

In order to connect a device, you need to download the following:

A certificate for this thing	208f60eb4f.cert.pem	<a href="#">Download</a>
A public key	208f60eb4f.public.key	<a href="#">Download</a>
A private key	208f60eb4f.private.key	<a href="#">Download</a>

**Download all keys and root CA**

You also need to download a root CA for AWS IoT:

A root CA for AWS IoT [Download](#)

[Activate](#)

**Activate keys**

The keys and cert will be used later

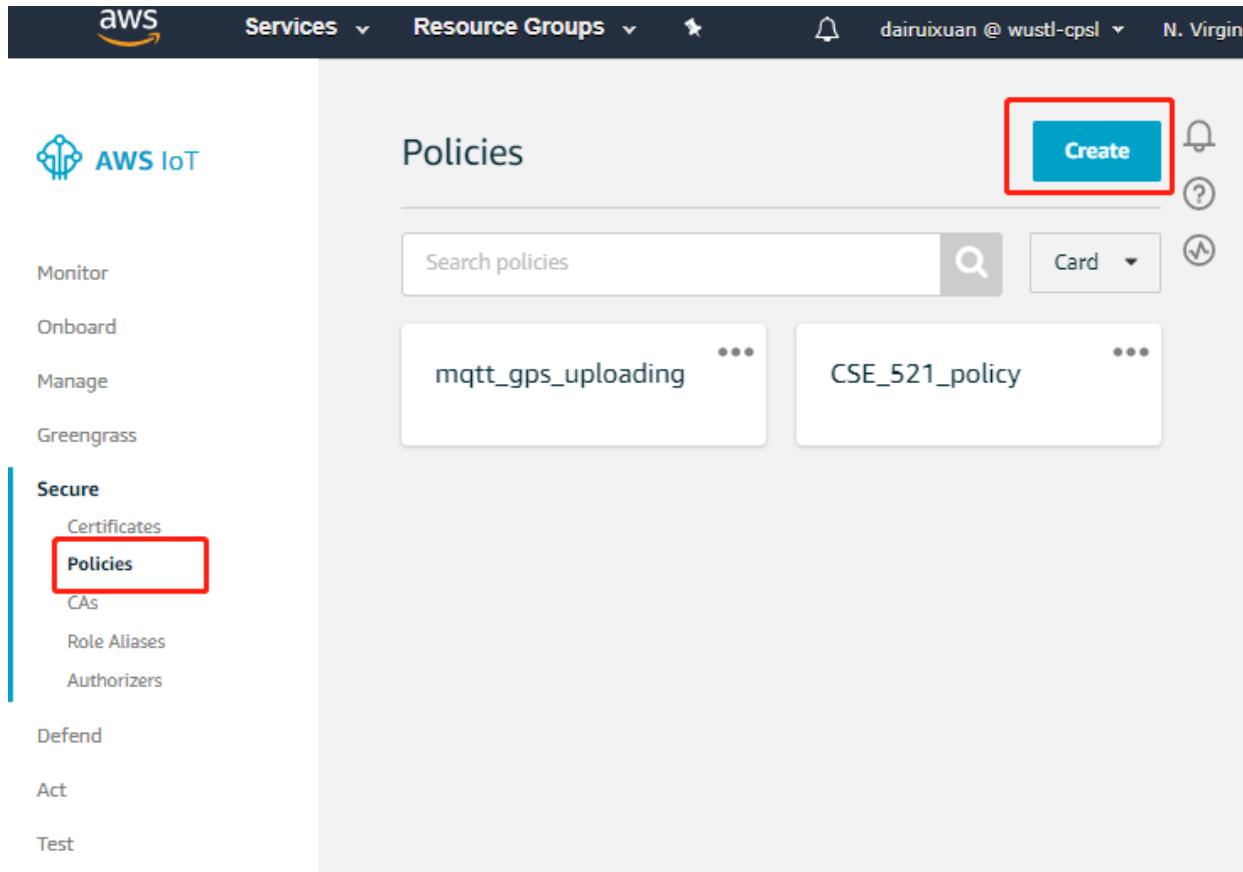
[Cancel](#)

[Done](#)

[Attach a policy](#)

# Create Policy

- A policy is attached to a key/cert
  - **It tells what this key/cert can do**



The screenshot shows the AWS IoT Policies interface. On the left, there's a sidebar with options like Monitor, Onboard, Manage, Greengrass, Secure (with Certificates, Policies highlighted with a red box, CAs, Role Aliases, Authorizers), Defend, Act, and Test. The main area is titled 'Policies' and contains a 'Create' button (also highlighted with a red box). Below it is a search bar and a 'Card' filter. Two policies are listed: 'mqtt\_gps\_uploading' and 'CSE\_521\_policy', each with a three-dot menu icon.



## Create a policy



Create a policy to define a set of authorized actions. You can authorize actions on one or more resources (things, topics, topic filters). To learn more about IoT policies go to the [AWS IoT Policies documentation page](#).

Name

CSE\_521\_Temp\_policy

### Add statements

Policy statements define the types of actions that can be performed by a resource.

[Advanced mode](#)

Action

iot:\*

Resource ARN

\*

Effect

Allow  Deny

[Remove](#)

[Add statement](#)

[Create](#)

AWS Services Resource Groups dairuixuan @ wustl-cpsl N. Virgin

AWS IoT Policies Create

Monitor Onboard Manage Greengrass

Secure Certificates Policies CAs Role Aliases Authorizers

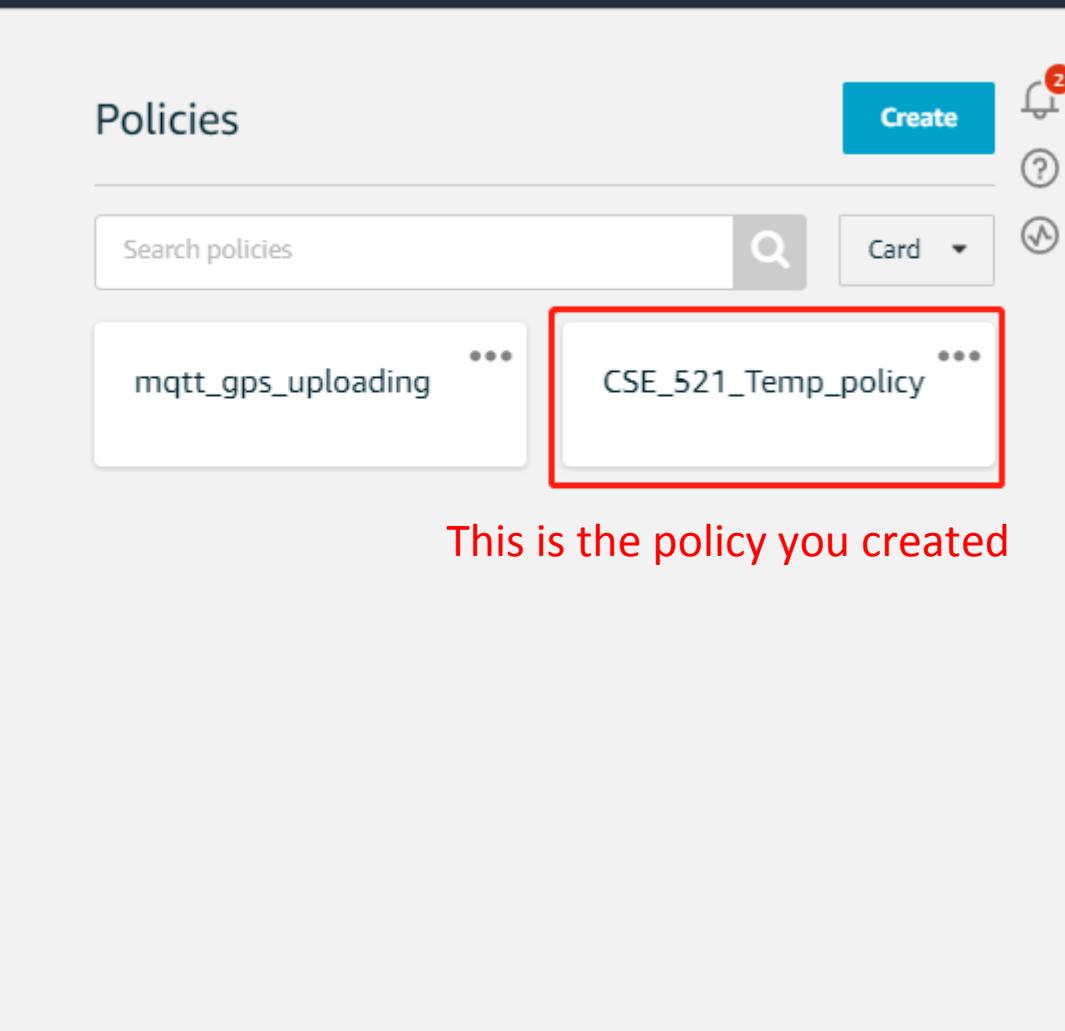
Defend Act Test

Search policies

mqtt\_gps\_uploading ...

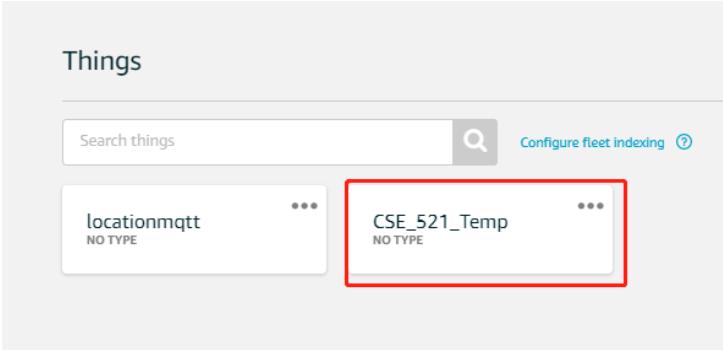
CSE\_521\_Temp\_policy ...

This is the policy you created



# Attach Policy

- Attach Policy to the key/cert
  - A policy tells what this key/cert can do



The screenshot shows the AWS IoT Things interface. On the left, under the 'Manage' section, the 'Things' option is selected. In the main area, there is a list of things: 'locationmqtt' and 'CSE\_521\_Temp'. The 'CSE\_521\_Temp' item is highlighted with a red box. A large blue arrow points from this screen to the right-hand details panel.

THING  
**CSE\_521\_Temp**  
NO TYPE

Details      Certificates

**Security**

Thing Groups      Create certificate      View other options

Billing Groups

Shadow

Interact

Activity

Jobs

Violations

Defender metrics

208f60eb4fab1b02f5...  
Click the cert

This is the key/cert you just created

CSE\_521\_Temp > 208f60eb4fab1b02f5d6...

CERTIFICATE  
**208f60eb4fab1b02f5d656963382b05cd5c690b481e710c6a3e899a...**  
INACTIVE

**Actions** ▾

- Activate
- Deactivate
- Revoke
- Accept transfer
- Reject transfer
- Revoke transfer
- Start transfer
- Attach policy**
- Attach thing
- Download
- Delete

**Details**

**Certificate ARN**

A certificate Amazon Resource Name (ARN) uniquely identifies this certificate.

`arn:aws:iot:us-east-1:006025899016:cert/208f60eb4fa...`

**Details**

**Issuer**  
OU=Amazon Web Services O\=Amazon.com Inc. L\=Seattle ST\=Washington C\=US

**Subject**  
CN=AWS IoT Certificate

**Create date**  
Aug 25, 2019 12:56:24 PM -0500

**Effective date**  
Aug 25, 2019 12:54:24 PM -0500

**Expiration date**  
Dec 31, 2049 5:59:59 PM -0600

## Attach policies to certificate(s)

Policies will be attached to the following certificate(s):

208f60eb4fab1b02f5d656963382b05cd5c690b481e710c6a3e899a73b5f9080

Choose one or more policies

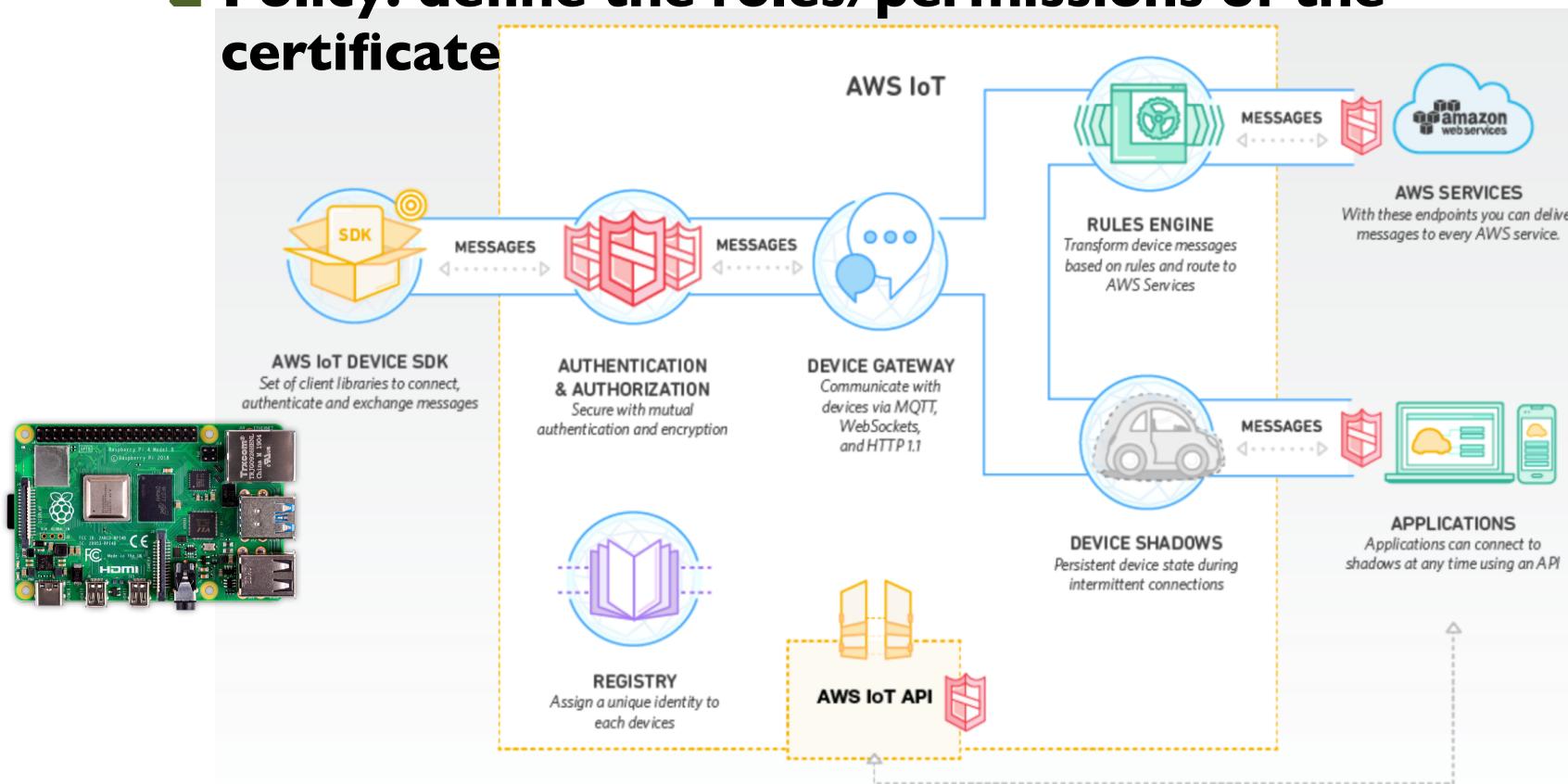
mqtt\_gps\_uploading [View](#)  
 CSE\_521\_Temp\_policy [View](#)

1 policy selected

OU=Amazon Web Services O=Amazon.com Inc. L=Seattle ST=Washington C=US  
Subject  
CN=AWS IoT Certificate

# AWS Things Summary

- ❑ **Shadow:** Store/retrieve some information
- ❑ **Certificate:** authenticate the device
- ❑ **Policy:** define the roles/permissions of the certificate



# Let's test it online!

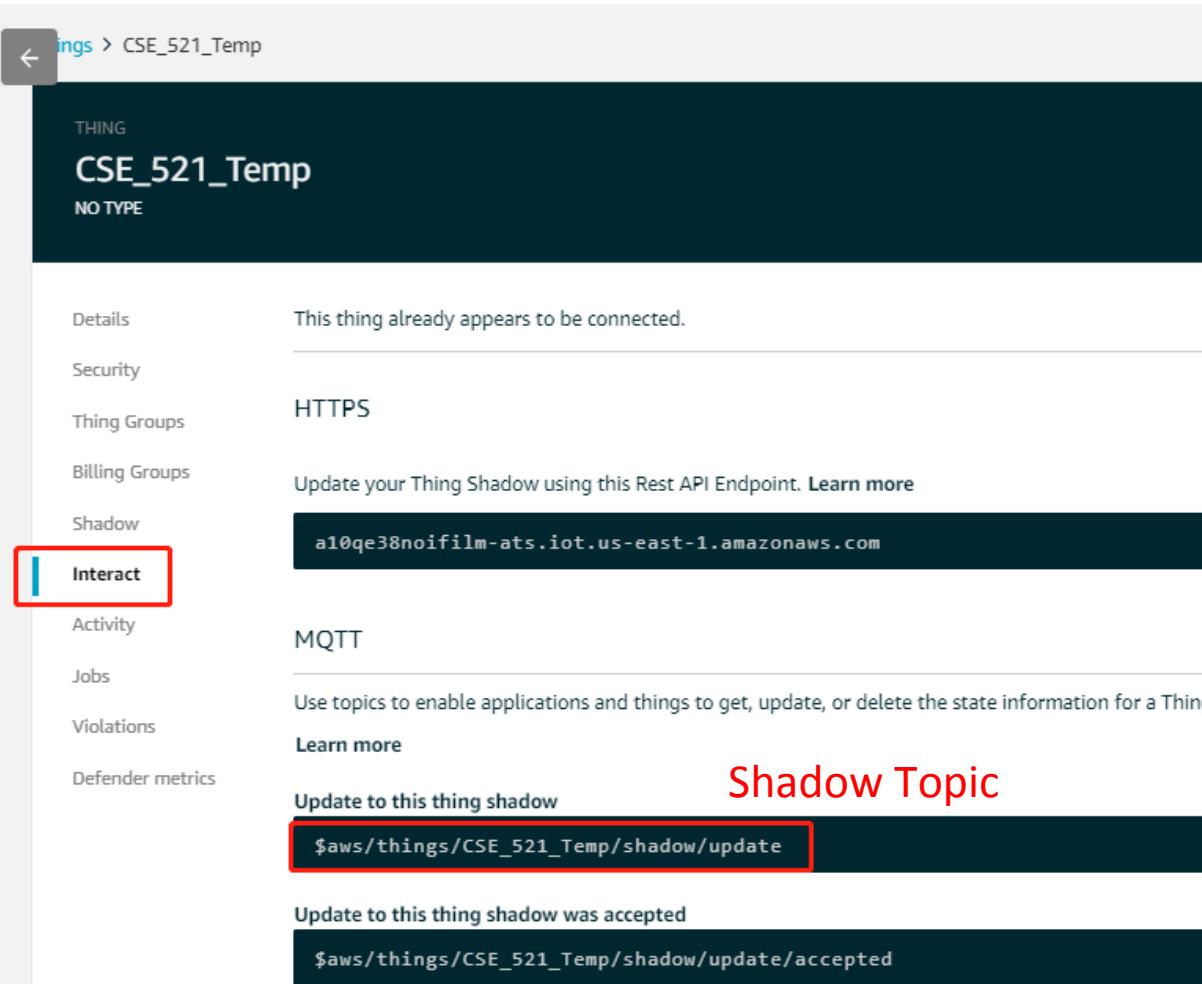
---



Washington University in St. Louis

# Basic Interact: Publish to the “Shadow”

- Get your “Shadow”
  - In your Thing Page



Things > CSE\_521\_Temp

THING

**CSE\_521\_Temp**

NO TYPE

Details This thing already appears to be connected.

Security

Thing Groups

Billing Groups

Shadow

**Interact**

Activity

Jobs

Violations

Defender metrics

HTTPS

Update your Thing Shadow using this Rest API Endpoint. [Learn more](#)

**a10qe38noifilm-ats.iot.us-east-1.amazonaws.com**

MQTT

Use topics to enable applications and things to get, update, or delete the state information for a Thing

[Learn more](#)

**Shadow Topic**

Update to this thing shadow

**\$aws/things/CSE\_521\_Temp/shadow/update**

Update to this thing shadow was accepted

**\$aws/things/CSE\_521\_Temp/shadow/update/accepted**



Monitor  
Onboard  
Manage  
Greengrass  
Secure  
Defend  
Act  
**Test**

## MQTT client ?

Connected as iotconsole-1566757232435-1

### Subscriptions

Subscribe to a topic

**Publish to a topic**

### Subscribe

Devices publish MQTT messages on topics. You can use this client to subscribe to a topic and receive these messages.

#### Subscription topic

Specify a topic to subscribe to, e.g. myTopic/1

Subscri...

#### Max message capture ?

100

#### Quality of Service ?

- 0 - This client will not acknowledge to the Device Gateway that messages are received
- 1 - This client will acknowledge to the Device Gateway that messages are received

#### MQTT payload display

- Auto-format JSON payloads (improves readability)
- Display payloads as strings (more accurate)
- Display raw payloads (in hexadecimal)

### Publish

Specify a topic and a message to publish with a QoS of 0.

\$aws/things/CSE\_521\_Temp/shadow/update

**Publish ...**

```
1 {  
2   "message": "Hello from AWS IoT console"  
3 }
```

Your Shadow  
Topic  
Topic Message

# Shadow Message

```
{ "state": {  
    "reported": {  
        "time": "13:45",  
        "temperature": "25"  
    },  
    "message": "Hello from AWS IoT console"  
}
```

A Shadow Message is a JSON object.

**Shadow message has strict formats.**

Please see

<https://docs.aws.amazon.com/iot/latest/developerguide/device-shadow-document-syntax.html>

# Update Shadow

## ➤ In your “Thing” Page

Things > CSE\_521\_Temp

THING  
**CSE\_521\_Temp**  
NO TYPE Actions ▾

Details      Shadow ARN

---

Security

Thing Groups

Billing Groups

**Shadow**  

Interact

Activity

Jobs

Violations

Defender metrics

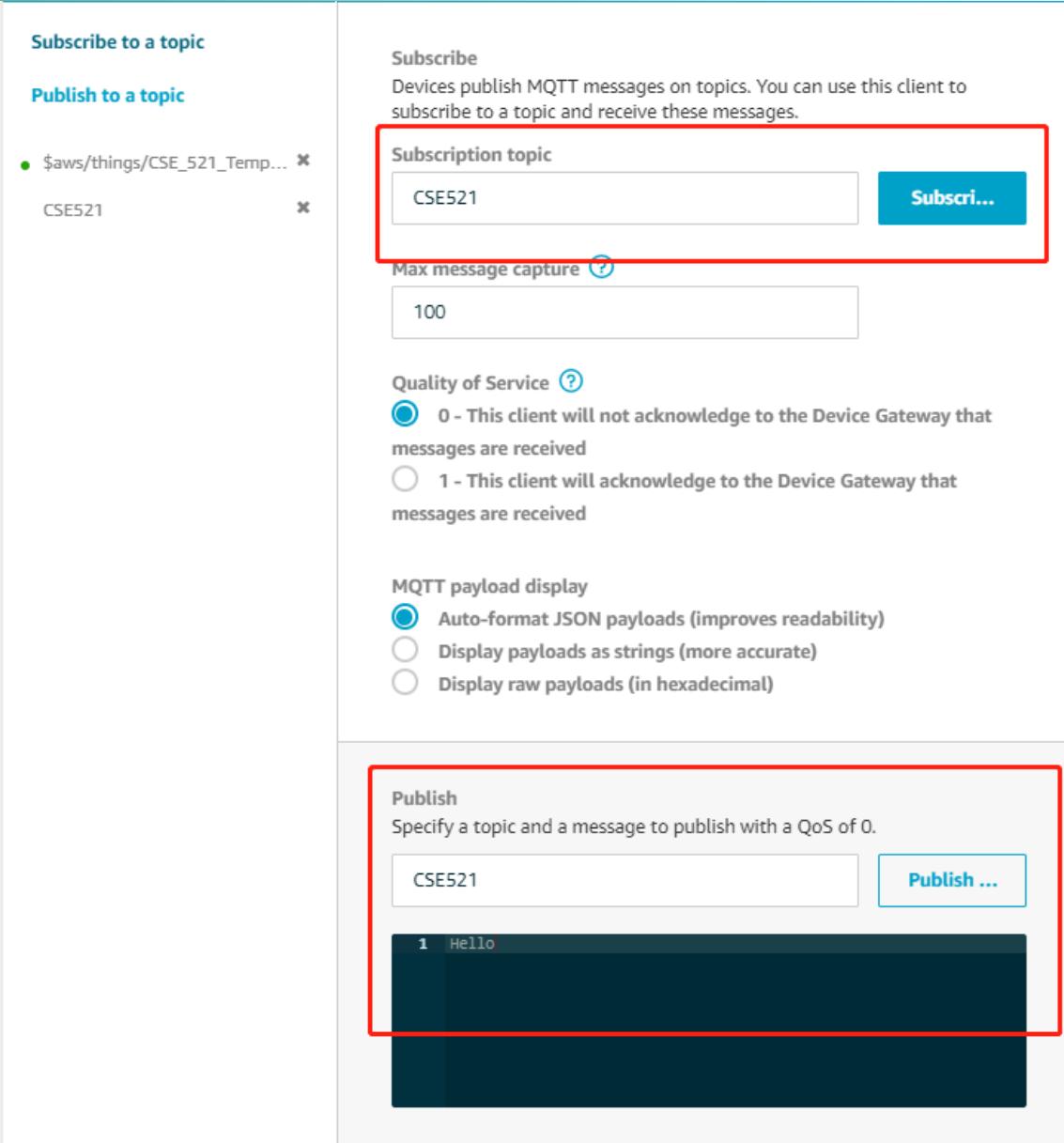
Shadow Document Delete Edit

Last update: Aug 25, 2019 1:33:35 PM -0500

**Shadow state:**

```
{  
  "reported": {  
    "time": "13:45",  
    "temperature": "25"  
  }  
}
```

# Basic Interact: Subscribe/Publish



The screenshot shows the AWS IoT Device Management console interface. On the left, there's a sidebar with 'Subscribe to a topic' and 'Publish to a topic' sections. Under 'Subscribe to a topic', there's a list with one item: '\$aws/things/CSE\_521\_Temp...' (with a CSE521 entry below it) and a red 'x'. Under 'Publish to a topic', there's a list with one item: 'CSE521' and a red 'x'.

**Subscribe**

Devices publish MQTT messages on topics. You can use this client to subscribe to a topic and receive these messages.

**Subscription topic**

 **Subscri...**

**Max message capture** ?

**Quality of Service** ?

- 0 - This client will not acknowledge to the Device Gateway that messages are received
- 1 - This client will acknowledge to the Device Gateway that messages are received

**MQTT payload display**

- Auto-format JSON payloads (improves readability)
- Display payloads as strings (more accurate)
- Display raw payloads (in hexadecimal)

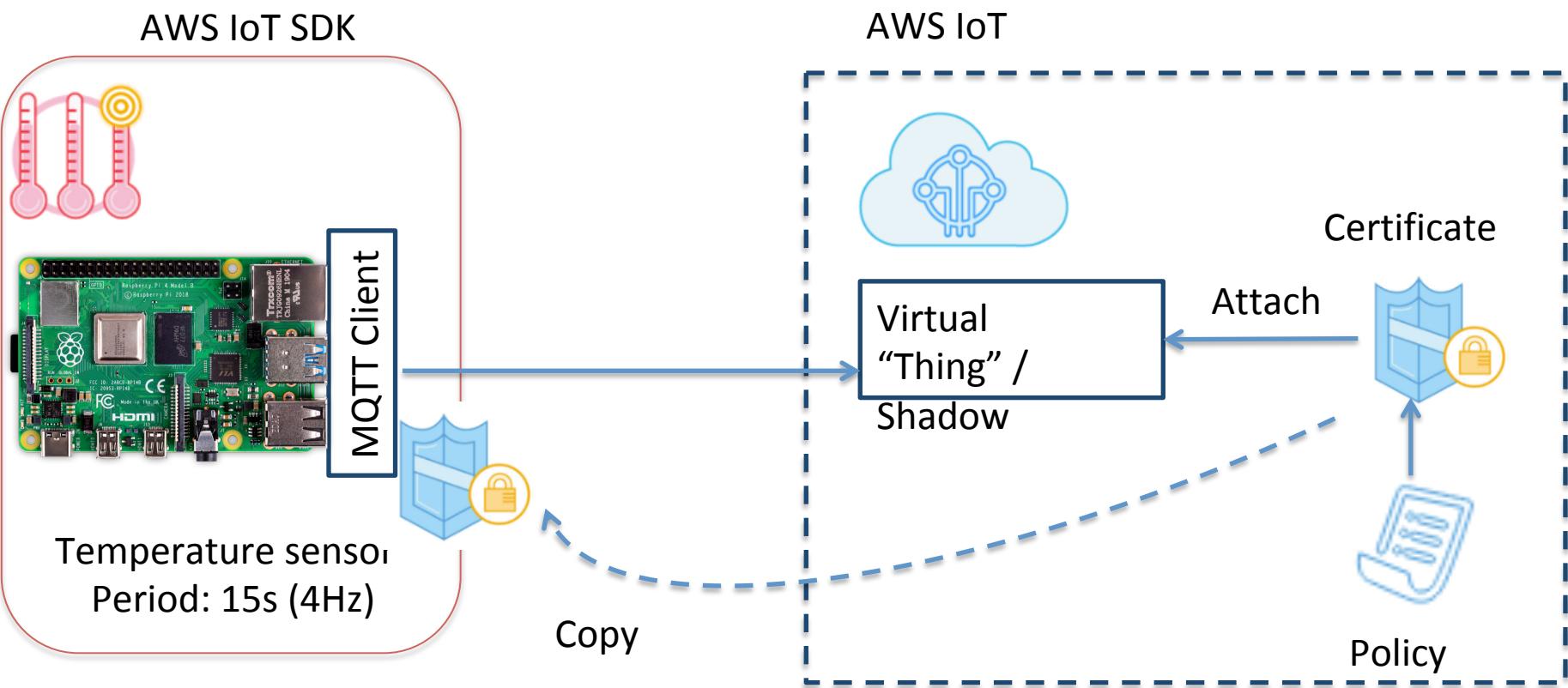
**Publish**

Specify a topic and a message to publish with a QoS of 0.

**CSE521** **Publish ...**

```
1 Hello
```

# Step 2: Connect a “Physical” Device



# Connect your Device

## ➤ Copy certificates to your **physical things**

- ❑ Note: through **scp** utility
- ❑ Downloaded before!

In order to connect a device, you need to download the following:

A certificate for this thing	208f60eb4f.cert.pem	<a href="#">Download</a>
A public key	208f60eb4f.public.key	<a href="#">Download</a>
A private key	208f60eb4f.private.key	<a href="#">Download</a>

[Download all keys and root CA](#)

You also need to download a root CA for AWS IoT:  
A root CA for AWS IoT [Download](#)

## ➤ Choose your **AWS SDK** (support MQTT)

- ❑ Node JS
- ❑ Python (pip install AWSIoTPythonSDK)
- ❑ Java
- ❑ ...

## ➤ Set up your client with SDK and the certificates



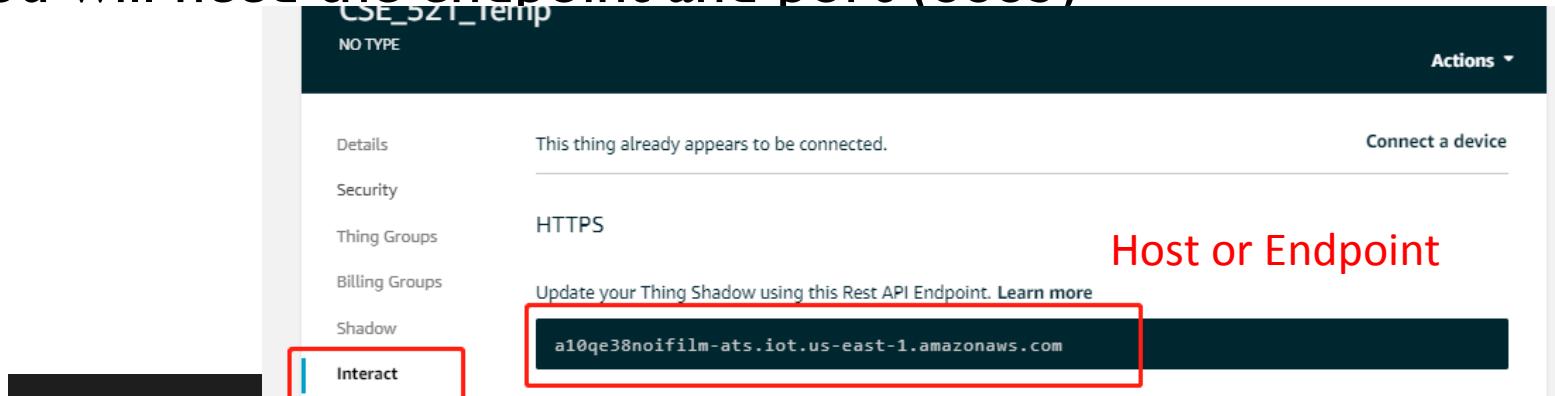
# Some Notes

## ➤ 1. Copy the certificates/keys

```
[ec2-user@ip-172-31-92-213 temp_demo]$  
[ec2-user@ip-172-31-92-213 temp_demo]$  
[ec2-user@ip-172-31-92-213 temp_demo]$ ls  
aws-iot-device-sdk-python demo.py  
CSE_521_Temp.cert.pem e556be14fc-certificate.pem.crt  
CSE_521_Temp.private.key e556be14fc-private.pem.key  
CSE_521_Temp.public.key e556be14fc-public.pem.key  
[ec2-user@ip-172-31-92-213 temp_demo]$
```

root-CA.crt  
**start.sh**

## ➤ 2. You will need the endpoint and port (8883)



```
host = "a10qe38noifilm-ats.iot.us-east-1.amazonaws.com" # Your thing's endpoint. See tutorial slides
rootCAPath = "root-CA.crt"
certificatePath = "e556be14fc-certificate.pem.crt"
privateKeyPath = "e556be14fc-private.pem.key"
port = 8883
clientId = "CSE521_temp"
topic = "$aws/things/CSE_521_Temp/shadow/update" # Shadow topic of your Thing
```

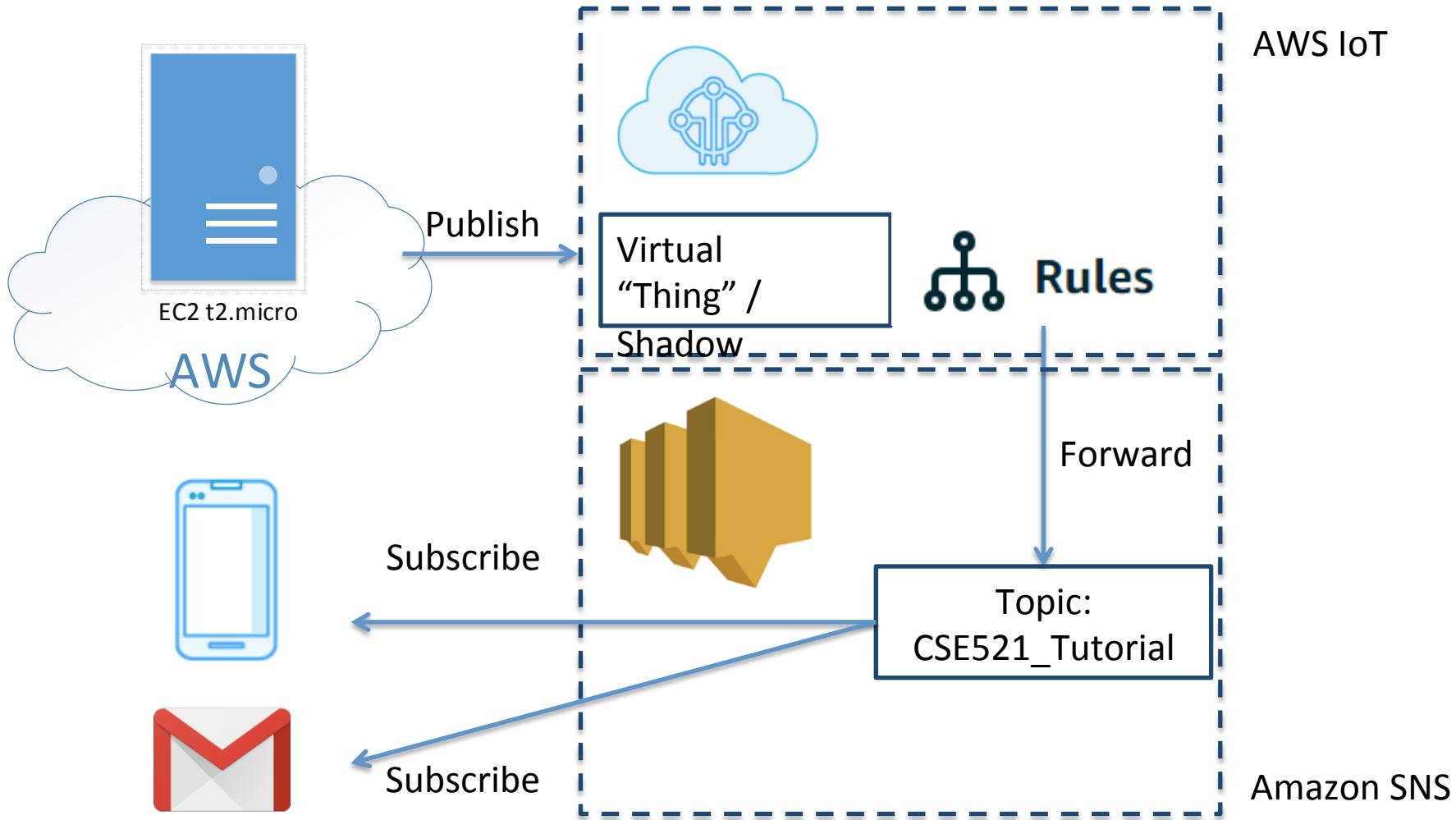
# SDK and Demo Codes

---

- <https://github.com/aws/aws-iot-device-sdk-python>
- <https://github.com/aws/aws-iot-device-sdk-python/blob/master/samples/basicPubSub/basicPubSub.py>

# More: Rule Engine, Link with SNS services

## ➤ Simple Notification Service



# Create a Rule in Amazon IoT

- Add a query to filter your interesting topic (event)

## Rule query statement

```
SELECT * FROM '$aws/things/RaspberryPi/shadow/update/accepted'
```

- Add an Action:

- Forward this message to SNS
- Specify Dest ARN
- Enable Rule

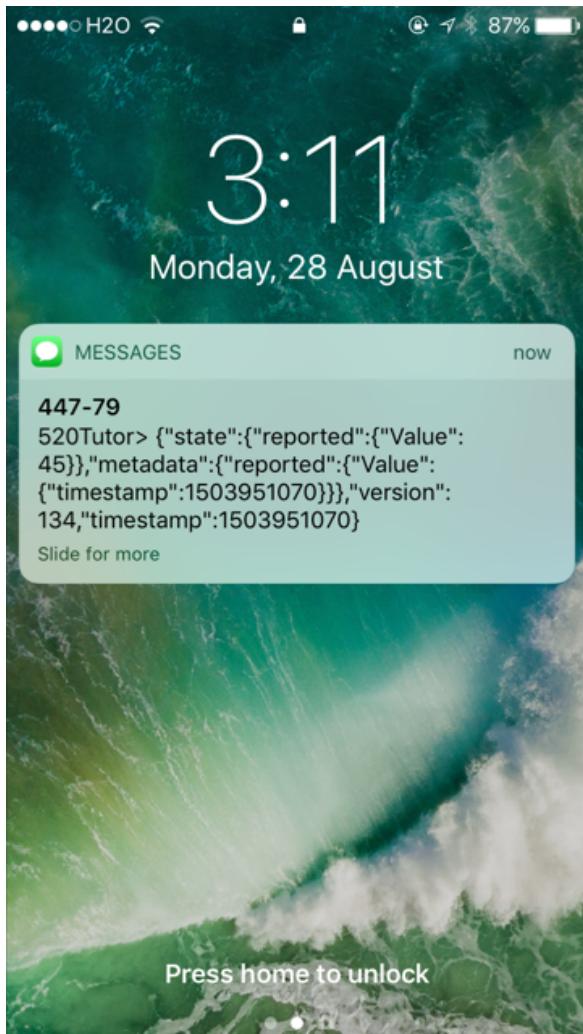
Configure action

 SNS Send a message as an SNS push notification

Rules

ForwardtoSMS  
ENABLED 

# Notification on SMS & Email



AWS Notification Message Inbox x  

 **520Tutor** no-reply@sns.amazonaws.com 3:11 PM (28 minutes ago) Star Reply Forward More

to me View

{"state":{"reported":{"Value":45}}, "metadata":{"reported":{"Value":{"timestamp":1503951070}}}, "version":134, "timestamp":1503951070}

--

If you wish to stop receiving notifications from this topic, please click or visit the [link](#) below to unsubscribe:  
[https://sns.us-west-2.amazonaws.com/unsubscribe.html?SubscriptionArn=arn:aws:sns:us-west-2:401317363811:CSE520S\\_Tutorial:00c54352-7d1a-4c09-9cc1-15aed3c395e3&Endpoint=naroahlee@gmail.com](https://sns.us-west-2.amazonaws.com/unsubscribe.html?SubscriptionArn=arn:aws:sns:us-west-2:401317363811:CSE520S_Tutorial:00c54352-7d1a-4c09-9cc1-15aed3c395e3&Endpoint=naroahlee@gmail.com)

Please do not reply directly to this email. If you have any questions or comments regarding this email, please contact us at  
<https://aws.amazon.com/support>

# One More Thing: Account Security

➤ DON'T UPLOAD YOUR KEY PUBLICLY!!!

Time to Open Source!



# What if... \$50,000 AWS Bill!

Quora

Ask or Search Quora

Ask Question

Fraud

Amazon Web Services

Amazon.com (product)

Hackers

+3



## My AWS account was hacked and I have a \$50,000 bill, how can I reduce the amount I need to pay?

For years, my bill was never above \$350/month on my single AWS instance. Then over the weekend someone got hold of my private key and launched hundreds of instances and racked up a \$50,000 bill before I found out about it on Tuesday. Amazon had sent a warning by email at \$15,000 saying they had found **our key posted publicly**, but I didn't see it. Naturally, this is a devastating amount of money to pay. I'm not saying I shouldn't pay anything, but this just a crazy amount in context. Amazon knew the account was compromised, that is why they sent an email, they knew the account history and I had only spent \$213 the previous month. I almost feel they deliberately let it ride to try to earn more money. Does anyone have any experience with this sort of problem?

# Thanks!

---

Ruixuan Dai

Aug. 29, 2019



Washington University in St. Louis