Cloud Tutorial: AWS IoT

CSE 520S Spring,
Jan. 16, 2020
Ruixuan Dai
XaaS: Basics in Cloud Computing
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 …
    - ![EC2](https://via.placeholder.com/150)
  - **PaaS (Platform as a Service):** execution runtime, middleware, web server, database, development tool …
    - ![AWS Lambda](https://via.placeholder.com/150)
    - ![AWS IoT Core](https://via.placeholder.com/150)
    - ![STADIA](https://via.placeholder.com/150)
  - **SaaS (Software as a Service):** email, virtual desktop, games …
    - ![Office 365](https://via.placeholder.com/150)
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: AWS IoT

The essence is **MESSAGING MIDDLEWARE**

Send messages between **sensors** and **servers**...
Software as a Service (SaaS)

- **SaaS**
  - You get a whole solution

- **Example**
  - Gmail
  - Dropbox
  - Office365

Service Provider Manages:

- APP
- Data
- Runtime
- Middleware
- OS
- Virtualization
- Server
- Storage
- Network
XaaS: A Recap

On-Premise
- APP
- Data
- Runtime
- Middleware
- OS
- Virtualization
- Server
- Storage
- Network

IaaS
- APP
- Data
- Runtime
- Middleware
- OS
- Virtualization

PaaS
- APP
- Data
- Runtime
- Middleware
- OS
- Virtualization

SaaS
- APP
- Data
- Runtime
- Middleware
- OS
- Virtualization

You Manage

Service Provider Manages
Tutorial: Hello! AWS IoT!!
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

Source: https://aws.amazon.com/iot-platform/
http://www.brain-smart.net/smart-brain-health-blog/page/2/#axzz4W4oSp8a6
Solution: AWS IoT

United: Connect + Communication
Stated: “Thing Shadow”

Smart: Other Cloud Service
Data Storage
Machine Learning

Source: https://aws.amazon.com/iot-platform/
Tutorial: Hello AWS IoT!

Temperature sensor
Period: 5s (0.2Hz)

AWS IoT

Publish
Forward
Subscribe

Amazon SNS

Message middleware

Source: https://aws.amazon.com/iot-platform/
Pointers

- **Amazon IoT**
  - [http://docs.aws.amazon.com/iot/latest/developerguide/what-is-aws-iot.html](http://docs.aws.amazon.com/iot/latest/developerguide/what-is-aws-iot.html)

- **Raspberry Pi**

- **Resource list** for course projects

- **Apply for $40 credits for Amazon AWS**
  - [https://aws.amazon.com/education/awseducate/apply/](https://aws.amazon.com/education/awseducate/apply/)
Get into AWS Manage Console

- Create your own AWS account
- Sign In IoT Manage Console

- [https://aws.amazon.com/iot/](https://aws.amazon.com/iot/)
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

- Shadow is a special topic in AWS IoT

- Certificates and policy
  - Authentication, Security
  - Permission and roles

Certificates -> your ID
Policy -> your permission book
Create a thing

1. AWS IoT Menu
   • Things ➔ Create

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

Name

cseS2q

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

No type selected

Create a type

Add this thing to a group

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

Thing Group

Groups /

Create group Change

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

Provide an attribute key, e.g. Manufacturer

Value

Provide an attribute value, e.g. Acme-Corporation

Add another

Clear

Show thing shadow

Cancel

Back

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

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

Skip certificate and create thing
You will need to add a certificate to your thing later before your device can connect to AWS IoT.
The keys and cert will be used later.
Create Policy

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

![AWS IoT Policies](image)
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**

cse520_policy

**Add statements**

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

**Action**

- `iot:*`

**Resource ARN**

- `*`

**Effect**

- **Allow**
- **Deny**

- **Create**
This is the policy you created
Attach Policy

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

Click the cert

This is the key/cert you just created
Attach policies to certificate(s)

Policies will be attached to the following certificate(s):
fa474a4756fb5e5cefd1aefcd9411b90344d96a8e38e28c2b2f50e9cb18d3ed2

Choose one or more policies

- mqtt_gps_uploading
- RuiPi_policy
- cse520_policy

1 policy selected
Cancel  Attach
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!
Basic Interact: Publish to the “Shadow”

- Get your “Shadow”
  - In your Thing Page

THING

cse520
NO TYPE

Details
Security
Thing Groups
Billing Groups

Shadow

Shadow ARN

A shadow ARN uniquely identifies the shadow for this thing. Learn more

arn:aws:iot:us-east-1:00602589016:thing/cse520

Shadow Document

Last update: Dec 31, 1969 6:00:00 PM -0600

Shadow state:

{}

Metadata:

{
  "metadata": {},
  "timestamp": 1579110725,
  "version": 3
}
Find your “Shadow” Topic

- **Topic:** can be seen as the “address”

---

**Thing:**
- **cse520**
- **NO TYPE**

**Details:**
- This thing already appears to be connected.

**Security:**
- HTTPS

**Thing Groups:**
- Update your Thing Shadow using this Rest API Endpoint. Learn more
  - a10qe38noifilm-ats.iot.us-east-1.amazonaws.com

**Billing Groups:**
- MQTT
  - Use topics to enable applications and things to get, update, or delete the state information. Learn more

**Shadow:**
- Update to this thing shadow
  - $aws/things/cse520/shadow/update

**Interact:**
- Update to this thing shadow was accepted
  - $aws/things/cse520/shadow/update/accepted
Using Embedded **Test Client** to Test

- In AWS IoT Page

![Subscription topic]

Subscribe to a topic

Publish to a topic

- $aws/things/cse520/shadow...

![Subscription topic explanation]

Subscription topic

- $aws/things/cse520/shadow/update

Max message capture

- 100

Quality of Service

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

MQTT payload display

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

![Publish topic]

Publish

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

- $aws/things/cse520/shadow/update

![Publish topic message]

Your Shadow Topic

Topic Message
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

Shadow ARN

A shadow ARN uniquely identifies the shadow for this thing. Learn more

```
arn:aws:iot:us-east-1:006025899016:thing/cse520
```

Shadow Document

Last update: Jan 15, 2020 11:56:23 AM -0600

Shadow state:

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

Metadata:
Basic Interact: Subscribe/Publish

- You can define your own Topic
- Once you have a subscriber that is subscribed to the topic, the subscriber can receive the message
Basic Interact: Subscribe/Publish

- You can define your own Topic
- Once you have a subscriber that is subscribed to the topic, the subscriber can receive the message

I am going to publish a tweet. “Today is beautiful”

Received message from Trump: “Today is beautiful”

Follow “subscribe”

Publisher

Subscriber
Step 2: Connect a “Physical” Device

Temperature sensor
Period: 15s (4Hz)

AWS IoT SDK

AWS SDK Client

AWS IoT

Virtual “Thing” / Shadow

Certificate
Attach
Policy
Copy
Connect your Device

- Copy certificates to your **physical things**
  - Note: through `scp` utility
  - Downloaded before!

- 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 to your real thing

   ```
   pi@RuiPi:~/cse520_demo $ ls -l
   total 20
   -rw-r--r-- 1 pi pi 3414 Jan 15 12:17 demo.py
   -rw-r--r-- 1 pi pi 1224 Jan 15 12:17 fa47fa4756-certificate.pem.crt
   -rw-r--r-- 1 pi pi 1675 Jan 15 12:17 fa47fa4756-private.pem.key
   -rw-r--r-- 1 pi pi 451 Jan 15 12:17 fa47fa4756-public.pem.key
   -rw-r--r-- 1 pi pi 1188 Jan 15 12:17 root-CA.crt
   ```

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

   ```
   host = "a10qe38nofilm-at.s.1ot.us-east-1.amazonaws.com" # Your thing's endpoint. See tutorial slides
   rootCAPath = "root-CA.crt"
   certificatePath = "fa47fa4756-certificate.pem.crt"
   privateKeyPath = "fa47fa4756-private.pem.key"
   port = 8883
   clientId = "CSE520"
   topic = "$aws/things/cse520/shadow/update" # Shadow topic of your Thing
   ```
SDK and Demo Codes

More: Rule Engine, Link with SNS services

Simple Notification Service

- Publish from AWS
- Subscribe to Amazon SNS
- Virtual "Thing" / Shadow
- Forward to Rules
- Topic: CSE521_Tutorial
Create a Rule in Amazon IoT

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

```sql
SELECT <Attribute> FROM <Topic Filter> WHERE <Condition>. For example: SELECT
learn more, see AWS IoT SQL Reference.
```

- Add an Action:
  - Forward this message to SNS
  - Specify Dest ARN
  - Enable Rule
Notification on SMS & Email

Monday, 28 August

447-79
520Tutor> {"state":{"reported":{"Value":45}},"metadata":{"reported":{"Value":{"timestamp":1503951070}},"version":134,"timestamp":1503951070}}

Press home to unlock

---

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

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!


1/16/20
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?
Project 0: AWS IoT

- Create Your AWS account
- Setup your things with certificates and policy
- Publish and subscribe a topic
  - Update your thing’s Shadow with AWS IoT Test Console
    - See tutorial
  - Use your computer as a “Thing” to update your thing’s Shadow
    - Install AWS IoT SDK, copy the certs, modify the code
    - Publish message to your Shadow Topic from your computer
- Publish and Subscribe on your computer
  - Publish a message to a topic in one client from your PC
  - Subscribe to the same topic in another client from your PC
  - Hints: the “clientID” cannot be the same for the two client. The two clients can share the same certs. You can use the shadow as the topic.
  - (You should see one terminal sends massage, and another terminal receives that message)
- Email your results (inline with some screenshots, you don’t need to write a report) to dairuixuan@wustl.edu
Some project examples

- Gesture recognition with smartwatch
  - Recognize the specific gesture to control the light

- Smart pet feeder
  - Food dispenser with schedules and smart control

- Smart mirror
  - Show personalized info in the mirror

Sensing, Connecting, Smart

If you have any question about the project, feel free to send me an Email
Thanks!

Ruixuan Dai