Cloud Tutorial: AWS EC2 and AWS IoT

TA for class CSE 520S, Fall, Aug/30/2017
Haoran Li
Agenda

- AWS EC2: IaaS
  - Project 0: Part 1

- AWS IoT: A Platform-as-a-Service example
  - Project 0: Part 2
Pointers

- Amazon EC2

- 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)

- 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/)
Today’s “Mainframe”

- Mainframe Computer (1970s)
  - Multiuser shares one Computer

- Datacenter (2016)
  - Multiuser shares multiple computers (A datacenter)

Service Provider Handles the Hardware.
You consume the computational resource.

Infrastructure as a Service (IaaS)

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

- Example
  - AWS EC2
  - AWS HPC

- Use case
  - Build up your VM cluster
Tutorial: Big Picture

- Create
  - Two EC2 Ubuntu Instances
    - T2.micro
  - Ensure the connectivity

- Micro benchmark
  - Latency
    - ping
  - Bandwidth
    - iperf3
Step 1: Create/Launch Instances

- USE AWS Console WebUI
- Create t2.micro Instance
  - 1 Xeon vCPU @ 2.5GHz
  - 1GB Memory
  - 8GB EBS Storage
  - ??? Network
- Create Operating System
  - AMI: Ubuntu 16.04 Server
- Get its Public DNS (HostName)
- Get the Private Ke

Source: https://www.shareicon.net/tag/network?cl=darkslategray&s=glyph
Access AWS EC2 Service

- **Sign Up and Sign In**

- **Choose EC2 Service**

Source: [https://aws.amazon.com/console](https://aws.amazon.com/console)
Launch instances

Launch instance in EC2 Dashboard

Resources

You are using the following Amazon EC2 resources in the US West (Oregon) region:

- 0 Running Instances
- 0 Dedicated Hosts
- 0 Volumes
- 1 Key Pairs
- 0 Placement Groups
- 0 Elastic IPs
- 0 Snapshots
- 0 Load Balancers
- 2 Security Groups

Just need a simple virtual private server? Get everything you need to jumpstart your project - compute, storage, and networking – for a low, predictable price. Try Amazon Lightsail for free.

Create Instance

To start using Amazon EC2 you will want to launch a virtual server, known as an Amazon EC2 instance.
Choose AMI

Choose Your Image Type (Template Operating System)

Step 1: Choose an Amazon Machine Image (AMI)

- Red Hat Enterprise Linux 7.4 (HVM), SSD Volume Type - ami-9fa343e7
  - Red Hat Enterprise Linux version 7.4 (HVM), EBS General Purpose (SSD) Volume Type
  - Root device type: ebs, Virtualization type: hvm
  - 64-bit

- Ubuntu Server 16.04 LTS (HVM), SSD Volume Type - ami-6e1a0117
  - Ubuntu Server 10.04 LTS (HVM), EBS General Purpose (SSD) Volume Type. Support available from Canonical (http://www.ubuntu.com/cloud/services).
  - Root device type: ebs, Virtualization type: hvm
  - 64-bit

- Microsoft Windows Server 2016 Base - ami-aa3cddcd2
  - Microsoft Windows 2016 Datacenter edition. [English]
  - Root device type: ebs, Virtualization type: hvm
  - 64-bit
## Instance Type (“Hardware” of VM)

- Choose your Virtual CPU number and Memory
  - **T2.micro 1 vCPU + 1GB @ $0.012 per Hour (running state)**

### Table of Instance Types

<table>
<thead>
<tr>
<th>Family</th>
<th>Type</th>
<th>vCPUs</th>
<th>Memory (GiB)</th>
<th>Instance Storage (GB)</th>
<th>EBS-Optimized Available</th>
<th>Network Performance</th>
<th>IPv6 Support</th>
</tr>
</thead>
<tbody>
<tr>
<td>General purpose</td>
<td>t2.nano</td>
<td>1</td>
<td>0.5</td>
<td>EBS only</td>
<td>-</td>
<td>Low to Moderate</td>
<td>Yes</td>
</tr>
<tr>
<td>General purpose</td>
<td>t2.micro</td>
<td>1</td>
<td>1</td>
<td>EBS only</td>
<td>-</td>
<td>Low to Moderate</td>
<td>Yes</td>
</tr>
<tr>
<td>General purpose</td>
<td>t2.small</td>
<td>1</td>
<td>2</td>
<td>EBS only</td>
<td>-</td>
<td>Low to Moderate</td>
<td>Yes</td>
</tr>
<tr>
<td>General purpose</td>
<td>t2.medium</td>
<td>2</td>
<td>4</td>
<td>EBS only</td>
<td>-</td>
<td>Low to Moderate</td>
<td>Yes</td>
</tr>
<tr>
<td>General purpose</td>
<td>t2.large</td>
<td>2</td>
<td>8</td>
<td>EBS only</td>
<td>-</td>
<td>Low to Moderate</td>
<td>Yes</td>
</tr>
</tbody>
</table>

**ATTENTION:** {CPU, Mem, Net} and Storage(EBS) are charged Separately!
### Storage ("Hard Drive")

**Step 4: Add Storage**

Your instance will be launched with the following storage device settings. You can attach additional EBS volumes and instance store volumes to your instance, or edit the settings of the root volume. You can also attach additional EBS volumes after launching an instance, but not instance store volumes. [Learn more](#) about storage options in Amazon EC2.

<table>
<thead>
<tr>
<th>Volume Type</th>
<th>Device</th>
<th>Snapshot</th>
<th>Size (GiB)</th>
<th>Volume Type</th>
<th>IOPS</th>
<th>Throughput (MB/s)</th>
<th>Delete on Termination</th>
<th>Encrypted</th>
</tr>
</thead>
<tbody>
<tr>
<td>Root</td>
<td>/dev/sda1</td>
<td></td>
<td>8</td>
<td>General Purpose SSD (GP2)</td>
<td>100 / 3000</td>
<td>N/A</td>
<td>Yes</td>
<td>Not Encrypted</td>
</tr>
</tbody>
</table>

Add New Volume

Free tier eligible customers can get up to 30 GB of EBS General Purpose (SSD) or Magnetic storage. [Learn more](#) about free usage tier eligibility and usage restrictions.

- **Mount Point**: /dev/sda1
- **Default 8GB**: $0.10 per GB-month
Review & Launch: Generate Private Key

A key pair consists of a **public key** that AWS stores, and a **private key file** that you store. Together, they allow you to connect to your instance securely. For Windows AMIs, the private key file is required to obtain the password used to log into your instance. For Linux AMIs, the private key file allows you to securely SSH into your instance.

Note: The selected key pair will be added to the set of keys authorized for this instance. Learn more about removing existing key pairs from a public AMI.

Create a new key pair

Key pair name

Naroah_Test

Download Key Pair

You have to download the **private key file** (*.pem file) before you can continue. **Store it in a secure and accessible location.** You will not be able to download the file again after it’s created.

Private Key: Use it to login your VM
Check the Dashboard

- Check the State: Running
- Check the Public DNS

We will ssh into the instance via this address:

```
ec2-52-88-47-79.us-west-2.compute.amazonaws.com
```

```
ec2-34-208-244-67.us-west-2.compute.amazonaws.com
```
Step 2: SSH into the VM

- Use SSH and Key to login the VM
Change the Permission of the Private Key

- Change it to “owner read only”, i.e. Permission 400
  - `sudo chmod 400 {Your_Private_Key}.pem`
Login Into it

- Host Address: Public DNS
- Username: `ubuntu`[1]

```bash
ssh -i {Your_Private_Key} ubuntu@{Pub_DNS_Name}
```

Windows User: Use Putty as a SSH client[2]

Step 3: Latency and Bandwidth Test

What’s the RTT delay?  
What’s the Throughput?

AWS

EC2 t2.micro

Low to Moderate

Network Performance
Connectivity Setting

- Security Group Settings
- Edit Inbound Rules
Allow Inbound Connections

Rules

- Enable All TCP/UDP/ICMP
Test the latency

- Using Ping
  - Get RTT Min/Avg/Max/Dev

Ping statistics:

```
--- 172.31.43.134 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 3997ms
rtt min/avg/max/mdev = 0.363/0.416/0.478/0.046 ms
```
Test the Bandwidth

- Using iperf3
  - Install: `sudo apt-get install iperf3`
  - Server: `iperf3 -s`
  - Client: `iperf3 -c {server_ip}`

1Gbps Bandwidth
Step 4: Remember to Stop/Terminate

- All about Money
  - E.g. t2.micro 1-VCPU + 1GB Mem @ $0.012 per Hour (running state)[1]

Remember to Stop Your VM

➢ Either shutdown from CLI:
  - `sudo shutdown -h now`

➢ Or Stop your instances from WebUI:
Stopped: However, they will still charge you

- Charge for EBS Storage[1]

Why am I charged for Elastic Block Store (EBS) when all my instances are stopped?

Issue

I've stopped all my Amazon EC2 instances, but I'm still receiving charges for EBS storage. How can I stop EBS charges accruing on my account for instances I'm not using?

- Pricing
  - E.g. gp2 volume: $0.10 per GB-month

Stopped V.S. Terminated

- “Terminate" your instance: PERMANENTLY DELETES the instance, and free the EBS storage.
- “Stop" the instance: Release the run time computational resource, but keep the disk.
- **Note:** Frequently committing your work to your repository.

<table>
<thead>
<tr>
<th>State</th>
<th>Stop</th>
<th>Terminate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Root volume (e.g. /dev/sda1)</td>
<td>The volume(EBS) is <strong>preserved</strong></td>
<td>The volume(EBS) is <strong>deleted</strong>.</td>
</tr>
<tr>
<td>Billing</td>
<td>You stop incurring charges for an instance as soon as its state changes to stopping.</td>
<td>You stop incurring charges for an instance as soon as its state changes to shutting-down.</td>
</tr>
</tbody>
</table>

More info: Read “**Differences Between Reboot, Stop, and Terminate**” in [1]

Create Your AWS account

Build up two EC2 instances and Setting up the Connectivity

Run ping / iperf3 and get latency / bandwidth statistics

1. Repeat this experiment (inbound bandwidth test)
2. Create Instance in different zone (e.g. us-west-2a/b/c), test the bandwidth between your PC and the EC2 Instance
   - Which Zone shows the best bandwidth / latency?
3. Pick up one “Zone”, write a script to repeat test every 2 hours (or more frequent, if you want) in a consecutive 24 hours
   - Plot a figure to show the latency against time
   - Plot a figure to show the bandwidth against time
4. Run a single test for 10 minute, will the stats change?
   - Note: iperf3 –c {server_ip} –t {seconds}

Email your results (inline, you don’t need to write a report) to lihaoran@email.wustl.edu
AWS IoT: A PaaS Example
Platform as a Service (PaaS)

- **PaaS**
  - You get a framework
  - Host Application
  - Tools

- **Example**
  - AWS IoT

- **Usecase**
  - Build up you’re smart A/C controller
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

Smart: Other Cloud Service
Data Storage
Machine Learning

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

Random Integer [1, 100]

EC2 t2.micro

AWS

PUSH

Publish

AWS IoT

Forward

Amazon SNS

Source: https://aws.amazon.com/iot-platform/
Step 1: Create a Virtual "Thing"

AWS IoT

Virtual “Thing” / Shadow
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/)
Create a thing

1. AWS IoT Menu
   - Registry
     - Things ➔ Create

2. Give a name
Basic Interact: Publish

- **Using Embedded MQTT Client to Test**

- **Check the Things Shadow**

8/30/17
Basic Interact: Subscribe

Devices publish MQTT messages on topics. Subscribe to a topic to view the messages published to it.

Subscription topic:
$saws/things/Test/shadow/update/accepted

Max message capture: 100
Quality of Service: 0

Subscribe to topic

MQTT client

Connected as iotconsole-1484713476597-4

Subscriptions

$aws/things/Test/shadow/update/accepted

$aws/things/Test/shadow/update/accepted

Jan 17, 2017 10:27:34 PM

Hide

{ 
  'state': 
  [ 
    'reported': 
    { 
      'Info': 'Hello AWS IoT!!' 
    } 
  ]
}
Step 2: Connect a “Physical” Device

EC2 t2.micro

Random Integer [1, 100]

MQTT Client

AWS

Copy

Virtual “Thing” / Shadow

AWS IoT

Certificate

Policy

Attach
Create and get Certificates

- Create Certificates
  - Security ➔ Certificates ➔ Create

- Download Cert Files
  - 1. public & private key
  - 2. thing cert
  - 3. Root CA for AWS

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

<table>
<thead>
<tr>
<th>Certificate Type</th>
<th>Filename</th>
<th>Download</th>
</tr>
</thead>
<tbody>
<tr>
<td>A certificate for this thing</td>
<td>f32c514adc.cert.pem</td>
<td>Download</td>
</tr>
<tr>
<td>A public key</td>
<td>f32c514adc.public.key</td>
<td>Download</td>
</tr>
<tr>
<td>A private key</td>
<td>f32c514adc.private.key</td>
<td>Download</td>
</tr>
</tbody>
</table>

You also need to download a root CA for AWS IoT from Symantec:
A root CA for AWS IoT [Download]
Create Policy and attach it to cert

- Create Policy
  - Security
    - Certificates
    - Policies
    - CAs

- Attach Policy to Certificates
Connect your Device

- Copy certificates to your EC2 Instance
  - Note: through `scp` utility

- Choose your AWS SDK (support MQTT)
  - Node JS
  - Python
  - Java

- You can also use third party MQTT tools
  - Python (paho-mqtt library)
Some Notes

1. You will need these certification when setting up the TLS1.2 verification

2. You will need the endpoint and port (8883) when connect to AWS IoT Gateway
More: Rule Engine, Link with SNS services

Simple Notification Service

AWS IoT

EC2 t2.micro

publish

Virtual “Thing” / Shadow

Rules

Forward

Topic: CSE520_Tutorial

Subscribe

Subscribe

Subscribe
Create a Rule in Amazon IoT

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

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

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

AWS Notification Message

520Tutor no-reply@sns.amazonaws.com 3:11 PM (28 minutes ago)

to me

{"state":{"reported":{"Value":45}},"metadata":{"reported":{"Value":45},"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:00c54352-7d4a-4c09-9cc1-15ae03c395e3&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
**AWS IoT: A PaaS Example**

Choose your Project topic:
Pick up the service you interest: Try it first.
And can you build your own service by using open source components?

**Publish**
EC2 t2.micro

**Subscribe**
EC2 t2.micro

**AWS IoT**

**Forward**
Amazon SNS

**AWS IoT as a MQTT Message Broker**

**AWS IoT storages Things States**

**AWS IoT provides the capability to link Against other services**

Source: https://aws.amazon.com/iot-platform/
Project 0 Part2: AWS IoT

- Create a Thing in AWS IoT
- Use Web MQTT tool to update Thing Shadow State
  - i.e. publish to shadow/update topic
- Setup MQTT client in EC2 instance
- Subscribe the shadow/update topic in EC2 instance
- Email the screenshots to lihaoran@email.wustl.edu
Project0: Part 2

- Sample Screenshot:
  - 1. Updated Shadow State in WebUI
  - 2. Printed message on EC2 terminal
Project 0:

- **Part 1 AWS EC2**
  - Bandwidth Test

- **Part 2 AWS IoT**
  - Subscribe to shadow/update

- Due on: 10AM Sep/6 (Wednesday)
Pointers

- **Amazon EC2**

- **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)

- **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/)
Thanks!

Haoran Li
Aug/30/2017
Project 0:

- Part 1 AWS EC2
  - Bandwidth Test

- Part 2 AWS IoT
  - Subscribe to shadow/update

- Due on: 10AM Sep/6 (Wednesday)
Project 0: Part 1 AWS EC2

- Create Your AWS account
- Build up two EC2 instances and Setting up the Connectivity
- Run ping / iperf3 and get latency / bandwidth statistics
  - 1. Repeat this experiment (inbound bandwidth test)
  - 2. Create Instance in different zone (e.g. us-west-2a/b/c), test the bandwidth between your PC and the EC2 Instance
    - **Which Zone** shows the best bandwidth / latency?
  - 3. Pick up one “Zone”, write a script to repeat test every 2 hours (or more frequent, if you want) in a consecutive 24 hours
    - **Plot a figure** to show the latency against time
    - **Plot a figure** to show the bandwidth against time
  - 4. Run a single test for 10 minute, will the stats change?
    - Note: `iperf3 -c {server_ip} -t {seconds}`
- Email your results (inline, you don’t need to write a report) to lihaoran@email.wustl.edu
Project 0 Part2: AWS IoT

- Create a Thing in AWS IoT
- Use Web MQTT tool to update Thing Shadow State
  - i.e. publish to shadow/update topic
- Setup MQTT client in EC2 instance
- Subscribe the shadow/update topic in EC2 instance
- Email the screenshots to lihaoran@email.wustl.edu
Project0: Part 2

- Sample Screenshot:
  - 1. Updated Shadow State in WebUI
  - 2. Printed message on EC2 terminal