Using Cloud Compute Services for Radiance Simulations

Andy McNeil
Disclaimers

• This presentation is offered with no warranty, use at your own risk.

• Andy is a hobbyist at best. Consult a network security expert for much better advice regarding security than what you’re about to receive.

• There is likely a better way to do everything I’m going to show you.
There is no cloud. It’s just someone else’s computer.
Why use someone else’s computer?

- Computing resources scale with needs
  - Add and drop instances as your workload changes
  - No need to coordinate with coworkers for running jobs
- Zero up front cost
- Track costs by project and bill computing expense to client
- Use Linux-only Radiance features without having to get a linux/mac computer
Andy’s experience in 2015-2016

As a contractor to Arup I (mostly) provide my own computing resources. For Radiance simulations I use the Amazon Elastic Compute Cloud (EC2). I’ve just been reviewing my usage statistics over the last 11 months, and thought I’d share with the lighting community.

With the EC2 you choose the specs of your instance when you request it. They have instances with specs ranging from 1 to 40 CPUs. I tended to use instances ranging from 32-40 cpus (depending on market pricing).

My usage statistics over the Past 11 months:
Total Cost: $888
Total CPU hours: 69,028
Cost per CPU hour: $0.013

On average over the 11 months I had 8.5 CPUs for $900, which is competitive with purchasing a high end desktop computer (assuming a 3-4 year life). But the real benefit is scalability. You see I didn’t have an 8 CPU computer, I had practically unlimited computing resources at my disposal when I needed them. For example, right now I have 4 x 40 CPU instances running (160 CPUs total!) and it is only costing me $1.40 an hour at spot market pricing.

- 11 months
- $900
- 69,028 CPU hours (equivalent to 8.5 constantly running CPUs)
- 160+ CPUs running for short periods
- Zero CPUs most of the time
Who’s Computer?

- AWS (Amazon)
- Azure (Microsoft)
- Google Cloud
- IBM Cloud
- Cloud & Heat - Distributed mini data centers that also provide heat to buildings
- And lots more…
Considerations - just a lazy list

• Access
• Storage
• Instance Configurations
• Computing Costs
• Spot / Low-Priority / Preemptible offering
• Your employer’s preference (and your client’s preference)
I chose AWS

• It was 2015 - AWS was the leader in cloud services

• Now things are pretty even between biggest cloud providers
  (but I still use AWS ‘cause I’m an old dog) 🐶
Getting Started with AWS in Twenty-one Easy Steps!

1. Create an AWS account
2. Choose a Region
3. Generate a key pair
4. Set up a security group
5. Choose spot or reserved instance
6. Select an instance type
7. Select a machine image
8. Launch instance
9. Connect to your instance
10. Install software
11. Save a machine image
12. Create persistent file storage
13. Mount persistent file storage
14. Upload Radiance model files
15. Start simulation
16. Set an alarm to terminate instance when simulation finishes
17. Launch an instance to retrieve results from file storage
18. Download results
19. Terminate instance
20. Get your cloud expenses and invoice your client
21. Bask in the glory of a job well done 😊
AWS Regions

• Choose a region based on proximity and cost.
  • N. California is the most expensive of the US regions, and typically the last to get new features.

https://www.concurrencylabs.com/blog/choose-your-aws-region-wisely/
Key Pairs

• Key pairs are used to securely access your AWS resources.

• Each key pair has a public and private component.
  • AWS keeps the public part
  • You get the private part.

• The private key is downloaded to your computer when it is created
  • There is no way to get the private key again
  • If you lose the private key file you’ll have to generate a new key pair and delete the old one
To start, you have no key pairs in your account.

Click: “Generate Key Pair”
Enter a name for your key pair and click create.
andy_ohio.pem is downloaded to my computer when I click “create”

Move your private key file somewhere secure and memorable. *I like to use a hidden folder for key files on my mac.*
Security Groups

- Sets access rules
  - restrict to IP address or only within security group

- Set accessible ports
  - SSH - port 22
  - NFS - port 2049
  - HTTP - port 80, HTTPS port 443

- When you launch an instance, a temporary security group is created, however it’s simplest to use one that’s already created.

- Useful for connecting EFS with ECS instances (we’ll get to this later)
Your account contains a default security group that allows inbound connections from other instances in the security group and nowhere else.

Click ‘Create Security Group’ to make a new security group.
We’ll create a security group with the following inbound access:

- **ssh from anywhere** (so you can connect to the instance from your computer)
- **nfs from anywhere** (we’ll restrict nfs to this security group in the next step)
Now we have two security groups.
Edit the inbound rules for the new security group by adding the ID of the security group in the source field for NFS. This will restrict inbound NFS to only services in this security group.
Now we have a custom security group.
Instance Types

• Dozens of machine instances available organized into categories:

• General Purpose
  • M - balanced compute, memory, and storage
  • T - burstable
  • A - ARM

• Compute Optimized
  • C: Lower cost per compute cycle

• Storage Optimized
  • I: large SSD local storage
  • D: Very large HDD local storage (up to 48 TB)
  • H: Large HDD local storage (up to 16TB) and balanced compute and memory

• Memory Optimized
  • R: more RAM per CPU
  • X: optimized for in memory database applications - lowest cost per GiB of RAM
  • Z: more RAM per CPU with highest CPU clock speed (4.0 GHz)

• Accelerated Computing (GPU FGPA)
  • P: General Purpose GPU
  • G: Graphics intensive GPU
  • F: FGPAs
Information about instance types

• Amazon’s info:
  https://aws.amazon.com/ec2-instance-types/

• This 3rd-party website is sortable, filterable and includes prices:
  https://www.ec2instances.info
Three ways to purchase an instance

- On-demand - regular
- Reserved - pay by the year
- Spot - pseudo auction
On-Demand Instance

• You start and stop the instance.
• You are billed per hour at fixed rate, only for the time you use.
• Nothing can stop the instance except you.
Reserved Instance

- Discounts on long term reservations. The instance is yours for the duration of the term whether you use it or not.
- 38% discount for a 1-year reservation paid monthly
- 72% discount for a 3-year reservation paid in advance

- You probably don’t want to use reserved instances
SPOT Instance

• You can get discounted rates on spare capacity (up to 90% discount).

• You bid the most you’re willing to pay per hour.

• You pay the market clearing rate each hour (not your bid rate).

• If the market clearing rate exceeds your bid amount your instance is terminated without warning.
Spot Instance Pricing History

Date
7/28/2019
3:45:04 PM UTC-0700

On-Demand price
$26.680

Availability Zone   Price
us-east-2a         $8.0064
us-east-2b         $26.680
My spot bidding strategy:
Bid a few cents more than the on-demand price.
Let’s launch an instance!
I’ve always used Amazon Linux. I don’t have a good reason.
Then you select the type of instance.

Make sure you only use the free tier eligible instance if you don’t want to pay.
Step 3: Configure Instance Details

Configure the instance to suit your requirements. You can launch multiple instances from the same AMI, request Spot instances to take advantage of the lower pricing, assign an access management role to the instance, and more.

### Number of instances
- **1**

### Purchasing option
- Request Spot instances

### Network
- **vpc-fed50057 (default)**

### Subnet
- **No preference (default subnet in any Availability Zone)**

### Auto-assign Public IP
- Use subnet setting (Enable)

### Placement group
- Add instance to placement group

### Capacity Reservation
- Open

### IAM role
- None

### Shutdown behavior
- Stop

### Enable termination protection
- Protect against accidental termination

### Monitoring
- Enable CloudWatch detailed monitoring
- Additional charges apply

### Tenancy
- Shared - Run a shared hardware instance
- Additional charges will apply for dedicated tenancy

### Elastic Inference
- Add an Elastic Inference accelerator
- Additional charges apply

### T2/T3 Unlimited
- Enable
- Additional charges may apply

### Advanced Details

---

*Nothing to do here yet.*

*But we’ll do some things here later.*
This is where you add EBS storage, if you decide to use that.
If you want to track costs by project, add a tag here.

Also, you need to activate the tag in the cost management page (we’ll do that towards the end)
We’ll use the security group we created.
Yes, the security group is open to the world, but they can’t get in without your private key.
Pick the key you want to use, and then acknowledge that you have the key file.
There it is, you did it!

You are clouding!
Connecting to your instance

- Mac & Linux: use ssh
- Windows: use putty or your favorite ssh client
Connecting on Mac / Linux

ssh command from mac

-i to use your private key file

ec2-user is the username for amazon linux

use the public IP for your instance
Connecting on Mac / Linux

Last login: Sun Aug 11 11:39:28 on ttys008
amcn:~ andy$ ssh -i aws_ohio.pem ec2-user@18.222.35.37
The authenticity of host '18.222.35.37 (18.222.35.37)' can't be established. ECDSA key fingerprint is SHA256:pxosq3oYz001HPIERekfuZUE0kLG/KnlG4AbWRXevuM. Are you sure you want to continue connecting (yes/no)?
yes you want to continue connecting.
whoops!
you need tighter permissions for your private key.
Connecting on Mac / Linux

chmod 600 to prevent access from any other user.
Connecting on Mac / Linux

ECDSA key fingerprint is SHA256:pxosq3oYz00IHPIERkfuZUE0kLG/KnlG4AbWRFXevuM.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '18.222.35.37' (ECDSA) to the list of known hosts.

Permissions 0644 for 'aws ohio.pem' are too open.
It is required that your private key files are NOT accessible by others.
This private key will be ignored.
Load key "aws ohio.pem": bad permissions
ec2-user@18.222.35.37: Permission denied (publickey,gssapi-keyex,gssapi-with-mic).

amcn:~ andy$ chmod 600 aws ohio.pem
amcn:~ andy$ ssh -i aws ohio.pem ec2-user@18.222.35.37

___|___|___
___|___|___|___
Amazon Linux 2 AMI

4 package(s) needed for security, out of 12 available
Run "sudo yum update" to apply all updates.
[ec2-user@ip-172-31-16-190 ~]$ 

then ssh again and viola!
Connecting on Windows

First you need to convert your private key file to a PuTTY private key file.

Open PuTTYgen
Connecting on Windows

Click “Load”

Browse to and select your pem file from amazon.
Connecting on Windows

Click “Save private key”
Connecting on Windows

Next Launch PuTTY
Connecting on Windows

Select Session, if not already selected

Enter the public IP address of your instance
Connecting on Windows

Select “Connection” -> “SSH” -> “Auth”

Click Browse, and select the private key ppk file.

Then click “Open”
Yes, you want to carry on connecting.
Connecting on Windows

at the “login as:” prompt enter ec2-user
Connecting on Windows

And you’re in!

Look at that screen. Your family is going to think you’re a nefarious hacker.
Regardless of OS, It’s the same from here…

- Bourne-Again Shell (bash)
- package manager: yum
Installing Software

- Amazon Linux uses yum package manager
  - First run `yum update` to get security and other updates
  - Then install packages needed to compile Radiance

```
sudo yum -y update
sudo yum -y install tcsh gcc gcc-c++ libX11-devel
```
# create a directory for Radiance
mkdir Radiance

cd Radiance

# download Radiance HEAD from radiance-online.org
wget --no-check-certificate http://www.radiance-online.org/software/snapshots/radiance-HEAD.tgz

# unpack tarballs

tar -xf radiance-HEAD.tgz
tar -xf radR52supp.tar.gz

# compile and install radiance

cd ray
sudo ./makeall install

# set raypath and copy cal files to raypath

echo RAYPATH=../usr/local/lib/ray/ > .bash_profile
echo export RAYPATH > .bash_profile
sudo cp src/cal/cal/* /usr/local/lib/ray/.

# check installation

rtrace -version
Run Mark Stock's Benchmark

# Install git
sudo yum -y install git

# clone benchmark repo
git clone https://github.com/markstock/Radiance-Benchmark4.git

# run the benchmark
export NCPU=16; make smp
# File Storage Options

<table>
<thead>
<tr>
<th>Service</th>
<th>price per GB</th>
<th>Description</th>
</tr>
</thead>
</table>
| Simple Storage Service (S3)   | $0.02        | • object storage in flat environment  
• good for archiving tarballs of old projects, but not for active storage                                                                                                                                 |
| Elastic Block Store (EBS)     | $0.10        | • Can be attached to a single EC2 instance  
• Easy to attach when launching instance  
• Limited to one availability zone (most regions have three zones)  
• Size is fixed and set when provisioned (you pay for empty GB)                                                                                     |
| Elastic File System (EFS)     | $0.30        | • Can be simultaneously attached to many EC2 instances  
• Must be mounted like a network drive (at the command line or with a startup script)  
• Spans availability zones (but limited to region)  
• Size is elastic, you only pay for GB used by your data                                                                                           |

Andy recommends EFS
Amazon Elastic File System (EFS)

Amazon EFS provides file storage for use with your EC2 instances.

Create file system

Getting started guide

Create

Create an Amazon EFS file system to store your files in the Amazon cloud. A file system grows and shrinks automatically with the files you put in, and you pay only for what you use.

Access

Write files to and read files from your Amazon EFS file system by using the NFSv4 protocol. Any number of EC2 instances can work with your file system at the same time, and your instances can be in multiple Availability Zones in a region.

Manage

You can easily administer your file system using the Amazon EFS console, CLI, and SDK.

Elastic File System documentation & support

Getting started guide | Documentation | Support | Forums
### Configure file system access

An Amazon EFS file system is accessed by EC2 instances running inside one of your VPCs. Instances connect to a file system by using a network interface called a mount target. Each mount target has an IP address, which we assign automatically or you can specify.

**VPC:** vpc-6a23697 (default)

### Create mount targets

Instances connect to a file system by using mount targets you create. We recommend creating a mount target in each of your VPC’s Availability Zones so that EC2 instances across your VPC can access the file system.

<table>
<thead>
<tr>
<th>Availability Zone</th>
<th>Subnet</th>
<th>IP address</th>
<th>Security groups</th>
</tr>
</thead>
<tbody>
<tr>
<td>us-east-2a</td>
<td>subnet-4b49428</td>
<td>Automatic</td>
<td>sg-67c3d05d0473dd - Basic</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>sg-6c4f04 - default</td>
</tr>
<tr>
<td>us-east-2b</td>
<td>subnet-3b52132</td>
<td>Automatic</td>
<td>sg-67c3d05d0473dd - Basic</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>sg-6c4f04 - default</td>
</tr>
<tr>
<td>us-east-2c</td>
<td>subnet-4b49428</td>
<td>Automatic</td>
<td>sg-67c3d05d0473dd - Basic</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>sg-6c4f04 - default</td>
</tr>
</tbody>
</table>
Lifecycle policy automatically moves files not recently accessed to lower cost storage. The next time its accessed there will be latency of up to 100ms, but then it's put back into the faster more expensive storage.
Your file system id is used for mounting to instances.

There are mount instructions on the EFS page if you ever forget. You don’t need to download this pdf again just for mounting instructions.

<table>
<thead>
<tr>
<th>VPC</th>
<th>Availability Zone</th>
<th>Subnet</th>
<th>IP address</th>
<th>Mount target ID</th>
<th>Network interface ID</th>
<th>Security groups</th>
<th>Mount target state</th>
</tr>
</thead>
<tbody>
<tr>
<td>us-east-2</td>
<td>us-east-2</td>
<td>subnet-6099dc04</td>
<td>172.31.46.223</td>
<td>front-99323e9</td>
<td>eni-0d81bc5245e38e4a</td>
<td></td>
<td>Creating</td>
</tr>
<tr>
<td>us-east-2</td>
<td>us-east-2</td>
<td>subnet-65521344</td>
<td>172.31.23.17</td>
<td>front-y6933</td>
<td>eni-0d81bc5245e38e4a</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Mounting Elastic File Storage

# Install tools
sudo yum install -y amazon-efs-utils

# create mount point
mkdir efs

# mount
sudo mount -t efs fs-8e7921f7:/ efs

# change owner and group if it’s your first time mounting the file system
sudo chown ec2-user efs
sudo chgrp ec2-user efs
Saving a custom machine image

- It’d be nice to not have to do all this setup every time, right?
- Machine Images - You can save the state of the machine
- Configuration script - Allows you to run updates and mount EFS drives when the instance is started
Machine Images

• Remember this step when launching an instance? ————>

Step 1: Choose an Amazon Machine Image (AMI)
An AMI is a template that contains the software configuration, operating system, application server, and applications required to launch your instance. You can select an AMI provided by AWS, our user community, or the AWS Marketplace, or you can select one of your own AMIs.

Quick Start

Amazon Linux 2 AMI (HVM), SSD Volume Type
- Select

Amazon Linux 2 AMI (HVM), SSD Volume Type
- Select

Red Hat Enterprise Linux 8 (HVM), SSD Volume Type
- Select

SUSE Linux Enterprise Server 15 SP1 (HVM), SSD Volume Type
- Select

Ubuntu Server 18.04 LTS (HVM), SSD Volume Type
- Select
Machine Images

• Let’s create a machine image!
Select the instance from which you want to generate a machine image.

Make sure it’s not busy, it will be rebooted to make the image!
Create Image

- Instance ID: i-0058e03059a0f87f5a
- Image name: Radiance
- Image description: HEAD 2019-08-11

Instance Volumes
- Volume Type: EBS
- Device: /dev/sdwa
- Snapshot: / snap-07068afe653e4666d
- Size (GB): 8
- Volume Type: General Purpose SSD
- IOPS: 100 / 3000
- Throughput (MB/s): N/A
- Delete on Termination: No
- Encrypted: Not Encrypted

Cancel  Create Image

Total size of EBS Volumes: 8 GB
When you create an EBS image, an EBS snapshot will also be created for each of the above volumes.
Select the instance from which you want to generate a machine image.

Make sure it’s not busy, it will be rebooted to make the image!
There’s our new image!
Configuration / Startup script

- There are some things you’ll want to run every time you start an instance, for example:
  - `sudo yum -y update`  
    (for security)
  - `sudo mount -t efs myfilesystem:/ efs`
- Configuration scripts can be provided as a text file when launching an instance.
- Or you could put this in a `@reboot` cron job on the machine image.
Example Configuration Script

aws_startup.bsh:

```bash
#!/bin/bash

yum -y update
mount -t efs fs-8e7921f7:/ /home/ec2-user/efs
```

When provided as a configuration script, it is run with root privileges, so you don’t need sudo.
When we go to launch an instance the new image is available.
I’m going to launch a 16 CPU instance this time.

NOT FREE TIER!
And let’s do a spot request too.
We can add our configuration script as a file under advanced details.
Tagging the spot request does not tag the instance. You need to tag the instance when it launches.
Step 6: Configure Security Group

A security group is a set of firewall rules that control the traffic for your instance. On this page, you can add rules to allow specific traffic to reach your instance. For example, if you want to set up a web server and allow internet traffic to reach your instance, add rules that allow unrestricted access to the HTTP and HTTPS ports. You can create a new security group or select from an existing one below. Learn more about Amazon EC2 security groups.

Assign a security group:  (Create a new security group)

*Select an existing security group

<table>
<thead>
<tr>
<th>Security Group ID</th>
<th>Name</th>
<th>Description</th>
<th>Actions</th>
</tr>
</thead>
<tbody>
<tr>
<td>sg-07c360b40753cd0d</td>
<td>Basic</td>
<td>SSS &amp; NFS</td>
<td>Copy to new</td>
</tr>
<tr>
<td>sg-0647c0d4</td>
<td>default</td>
<td>default VPC security group</td>
<td>Copy to new</td>
</tr>
</tbody>
</table>

**Warning**
Rules with source of 0.0.0.0/0 allow all IP addresses to access your instance. We recommend setting security group rules to allow access from known IP addresses only.

Inbound rules for sg-07c360b40753cd0d (Selected security groups: sg-07c360b40753cd0d)

<table>
<thead>
<tr>
<th>Type</th>
<th>Protocol</th>
<th>Port Range</th>
<th>Source</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SSH</td>
<td>TCP</td>
<td>22</td>
<td>0.0.0.0/0</td>
<td></td>
</tr>
<tr>
<td>SSH</td>
<td>TCP</td>
<td>22</td>
<td>::0</td>
<td></td>
</tr>
<tr>
<td>NFS</td>
<td>TCP</td>
<td>2049</td>
<td>sg-07c360b40753cd0d</td>
<td>Basic</td>
</tr>
</tbody>
</table>
Now that the spot request is fulfilled, you should remember to tag the resulting instance.
Now you should remember to tag your spot fulfilled instance.
Now you should remember to tag your spot fulfilled instance.
Now you should remember to tag your spot fulfilled instance.
Upload model files - Mac / Linux

- scp (secure copy)
- Copies files over ssh (similar syntax to ssh)

```
scp -i aws_ohio.pem -r my_model/ ec2-user@18.222.35.37:~/efs/
```

- `-i key.pem` : private key
- `-r` : recursive (copies directories and contents)
Upload model files - Windows

• `pscp` (PuTTY secure copy)
• Copies files over ssh

```bash
pscp -i aws_ohio.ppk -r my_model/ ec2-user@18.222.35.37:
```

-i key.ppk : private key
-r : recursive (copies directories and contents)
Uploading David’s example files

# These commands are run locally.
# upload zip files

cp -i ~/aws_ohio.pem 01_genBSDF.zip ec2-user@18.222.35.37:~/efs/.
cp -i ~/aws_ohio.pem example_noResults.zip ec2-user@18.222.35.37:~/efs/.
Start Simulation!!!!

- nohup (very important)
  - keeps a process running until it finishes, even if you log out
  - sends stdout to a file, nohup.out by default
  - to stop a process running with nohup, use `ps` to get the process id and `kill` to end it.
Running David’s BSDF example

# unpack zip files
unzip 01_genBSDF.zip
unzip example_noResults.zip

# change permission to make scripts executable
# this can be avoided if script has ‘bash myscript.sh’ instead of ‘./myscript.sh’

cd example_noResults
chmod +x *.sh scripts/* .sh

# copy cal file to current directory so script can find it

cp ../01_genBSDF/window7_2side.cal .

# run command

nohup time bash 00_all.sh &
### Benchmarks on David’s example

<table>
<thead>
<tr>
<th>Instance Type</th>
<th>vCPU</th>
<th>Physical Processor</th>
<th>Clock Speed</th>
<th>Memory (GiB)</th>
<th>File Storage</th>
<th>Time (s)</th>
<th>On Demand Rate/hr</th>
<th>Cost</th>
<th>Spot (18 Aug, 15:00) Rate/hr</th>
<th>Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>c5.24xlarge</td>
<td>96</td>
<td>2nd Gen Xeon Platinum 8175CL</td>
<td>3.0 GHz</td>
<td>192</td>
<td>EBS</td>
<td>263</td>
<td>$4.08</td>
<td>$0.298</td>
<td>$0.91</td>
<td>$0.066</td>
</tr>
<tr>
<td>c5.24xlarge</td>
<td>96</td>
<td>2nd Gen Xeon Platinum 8175CL</td>
<td>3.0 GHz</td>
<td>192</td>
<td>EFS</td>
<td>441</td>
<td>$4.08</td>
<td>$0.500</td>
<td>$0.91</td>
<td>$0.111</td>
</tr>
<tr>
<td>m5.24xlarge</td>
<td>96</td>
<td>Intel Xeon Platinum 8175</td>
<td>3.1 GHz</td>
<td>384</td>
<td>EBS</td>
<td>276</td>
<td>$4.61</td>
<td>$0.354</td>
<td>$0.96</td>
<td>$0.074</td>
</tr>
<tr>
<td>m5d.24xlarge</td>
<td>96</td>
<td>Intel Xeon Platinum 8175</td>
<td>3.1 GHz</td>
<td>384</td>
<td>SSD</td>
<td>278</td>
<td>$5.42</td>
<td>$0.419</td>
<td>$0.96</td>
<td>$0.074</td>
</tr>
<tr>
<td>m5d.24xlarge</td>
<td>96</td>
<td>Intel Xeon Platinum 8175</td>
<td>3.1 GHz</td>
<td>384</td>
<td>EFS</td>
<td>463</td>
<td>$5.42</td>
<td>$0.697</td>
<td>$0.96</td>
<td>$0.123</td>
</tr>
<tr>
<td>c5.18xlarge</td>
<td>72</td>
<td>Intel Xeon Platinum 8124M</td>
<td>3 GHz</td>
<td>144</td>
<td>EBS</td>
<td>309</td>
<td>$3.06</td>
<td>$0.263</td>
<td>$0.80</td>
<td>$0.069</td>
</tr>
<tr>
<td>c5.18xlarge</td>
<td>72</td>
<td>Intel Xeon Platinum 8124M</td>
<td>3 GHz</td>
<td>144</td>
<td>EBS</td>
<td>501</td>
<td>$3.06</td>
<td>$0.426</td>
<td>$0.80</td>
<td>$0.111</td>
</tr>
<tr>
<td>c5.12xlarge</td>
<td>48</td>
<td>2nd Gen Xeon Platinum 8175CL</td>
<td>3.0 GHz</td>
<td>96</td>
<td>EBS</td>
<td>464</td>
<td>$2.04</td>
<td>$0.263</td>
<td>$0.46</td>
<td>$0.059</td>
</tr>
<tr>
<td>c5.12xlarge</td>
<td>48</td>
<td>2nd Gen Xeon Platinum 8175CL</td>
<td>3.0 GHz</td>
<td>96</td>
<td>EFS</td>
<td>638</td>
<td>$2.04</td>
<td>$0.362</td>
<td>$0.46</td>
<td>$0.082</td>
</tr>
<tr>
<td>m5.12xlarge</td>
<td>48</td>
<td>Intel Xeon Platinum 8175</td>
<td>3.1 GHz</td>
<td>192</td>
<td>EBS</td>
<td>483</td>
<td>$2.06</td>
<td>$0.276</td>
<td>$0.54</td>
<td>$0.072</td>
</tr>
<tr>
<td>m5.12xlarge</td>
<td>48</td>
<td>Intel Xeon Platinum 8175</td>
<td>3.1 GHz</td>
<td>192</td>
<td>EFS</td>
<td>665</td>
<td>$2.06</td>
<td>$0.380</td>
<td>$0.54</td>
<td>$0.100</td>
</tr>
<tr>
<td>c5.9xlarge</td>
<td>36</td>
<td>Intel Xeon Platinum 8124M</td>
<td>3 GHz</td>
<td>72</td>
<td>EBS</td>
<td>592</td>
<td>$1.53</td>
<td>$0.252</td>
<td>$0.39</td>
<td>$0.064</td>
</tr>
<tr>
<td>c5.9xlarge</td>
<td>36</td>
<td>Intel Xeon Platinum 8124M</td>
<td>3 GHz</td>
<td>72</td>
<td>EFS</td>
<td>774</td>
<td>$1.53</td>
<td>$0.329</td>
<td>$0.39</td>
<td>$0.084</td>
</tr>
<tr>
<td>c5.4xlarge</td>
<td>16</td>
<td>Intel Xeon Platinum 8124M</td>
<td>3 GHz</td>
<td>32</td>
<td>EBS</td>
<td>1220</td>
<td>$0.68</td>
<td>$0.230</td>
<td>$0.15</td>
<td>$0.051</td>
</tr>
<tr>
<td>c5.4xlarge</td>
<td>16</td>
<td>Intel Xeon Platinum 8124M</td>
<td>3 GHz</td>
<td>32</td>
<td>EFS</td>
<td>1415</td>
<td>$0.68</td>
<td>$0.267</td>
<td>$0.15</td>
<td>$0.059</td>
</tr>
</tbody>
</table>
Compute optimized instances completed the script in less time and cost less.
Increasing CPU count has diminishing returns, but for this simulation up to 96 CPUs are still effective.
The simulation cost increases with more vCPUs.

Cost is hourly price time simulation time. This assumes the instance is terminated immediately. Spot prices subject to variability.
Using instance attached storage (EBS) for simulation files was faster than using network attached storage (EFS)

![Graph showing completion time in seconds versus number of CPUs (log scale)]
Alarms

- Alarms take an action when the instance usage crosses a threshold.

- For example (and the only reason I use alarms):
  - Terminate the instance when the CPU is below 1% for 10 minutes

- Alarms cost $0.10 each, though they save money by terminating idle instances (just make sure your results are on or moved to persistent storage).
Click on monitoring tab

Then on Create Alarm button
Alarms cost $0.10 each.

They save money by terminating idle instances (just make sure your results are on or moved to persistent storage).
Download Results

- If you used an alarm to terminate your instance, you’ll need to launch another instance to get your results.

- Use `scp` (Mac/Linux) or `pscp` (Windows) to download your result.

  ```bash
  scp -i aws_ohio.pem -r ec2-user@18.222.35.37:~/efs/my_model/results/ .
  pscp -i aws_ohio.ppk -r ec2-user@18.222.35.37:~/efs/my_model/results/ .
  ```

  copy this to here
Downloading Results

# These commands are run locally.
# download results file
scp -i ~/aws_ohio.pem -r ec2-user@18.222.35.37:~/efs/example_noResults/result .
Terminate Instance

• When you’re done, you terminate your instance

• *Not much else to say, this one is pretty self explanatory.*
Possibly the easiest step.
Invoice your client

• Activate tags for cost tracking

• Be diligent about tagging resources (this is the hard part)
  • Resources can NOT be tagged retroactively

• Use cost explorer to aggregate costs for each project
Click cost allocation tags
Select the tag(s) to track for cost allocations
Activate them
Click cost explorer button
Click explore costs
You can filter by many attributes, including tags.

And you can download a CSV.
Bask in the glory!
Andy's Cloud College

THIS DIPLOMA IS PRESENTED TO

You

For Clouding Along During the Cloud Course

21 August 2019