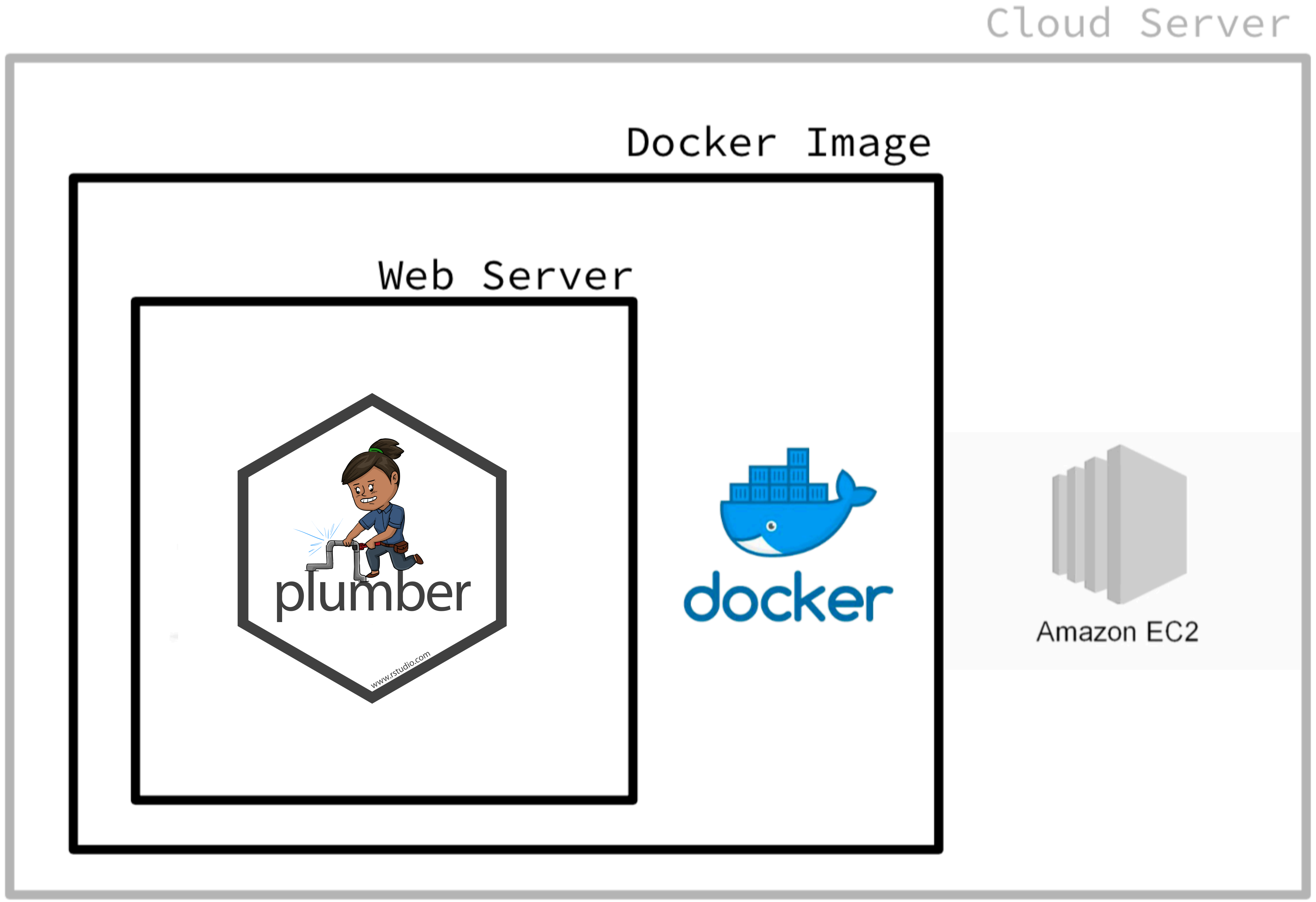
What are we going to build?



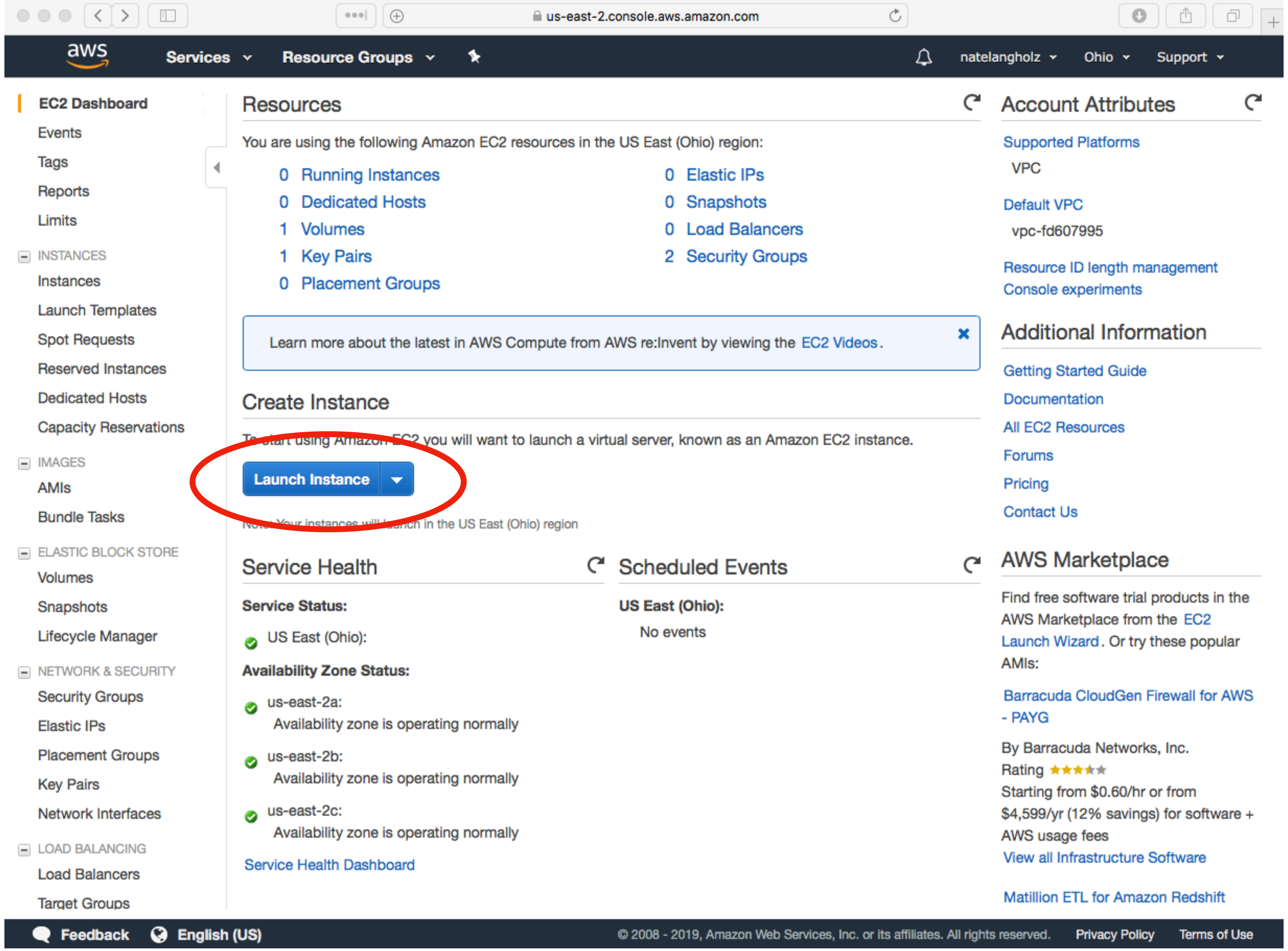
We will be able post json through a curl request in the same fashion that we did for our flask api

Let’s look at deploying it on the cloud so anyone can look at it

Start with an AWS ec2 instance. I would recommend creating a new instance to deploy this.

We can use the same .pem key (will probably need to move it but that’s fine)

`Launch` an instance (a reminder)



us-east-2.console.aws.amazon.com

aws Services Resource Groups natelangholz Ohio Support

EC2 Dashboard
Events
Tags
Reports
Limits

INSTANCES
Instances
Launch Templates
Spot Requests
Reserved Instances
Dedicated Hosts
Capacity Reservations

IMAGES
AMIs
Bundle Tasks

ELASTIC BLOCK STORE
Volumes
Snapshots
Lifecycle Manager

NETWORK & SECURITY
Security Groups
Elastic IPs
Placement Groups
Key Pairs
Network Interfaces

LOAD BALANCING
Load Balancers
Target Groups

Resources

You are using the following Amazon EC2 resources in the US East (Ohio) region:

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

Learn more about the latest in AWS Compute from AWS re:Invent by viewing the EC2 Videos.

Create Instance

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

Launch Instance

Note: Your instances will launch in the US East (Ohio) region

Service Health

Service Status:

✓ US East (Ohio):

Availability Zone Status:

✓ us-east-2a:
Availability zone is operating normally

✓ us-east-2b:
Availability zone is operating normally

✓ us-east-2c:
Availability zone is operating normally

Service Health Dashboard

Scheduled Events

US East (Ohio):

No events

Account Attributes

Supported Platforms
VPC

Default VPC
vpc-fd607995

Resource ID length management
Console experiments

Additional Information

Getting Started Guide
Documentation
All EC2 Resources
Forums
Pricing
Contact Us

AWS Marketplace

Find free software trial products in the AWS Marketplace from the EC2 Launch Wizard. Or try these popular AMIs:

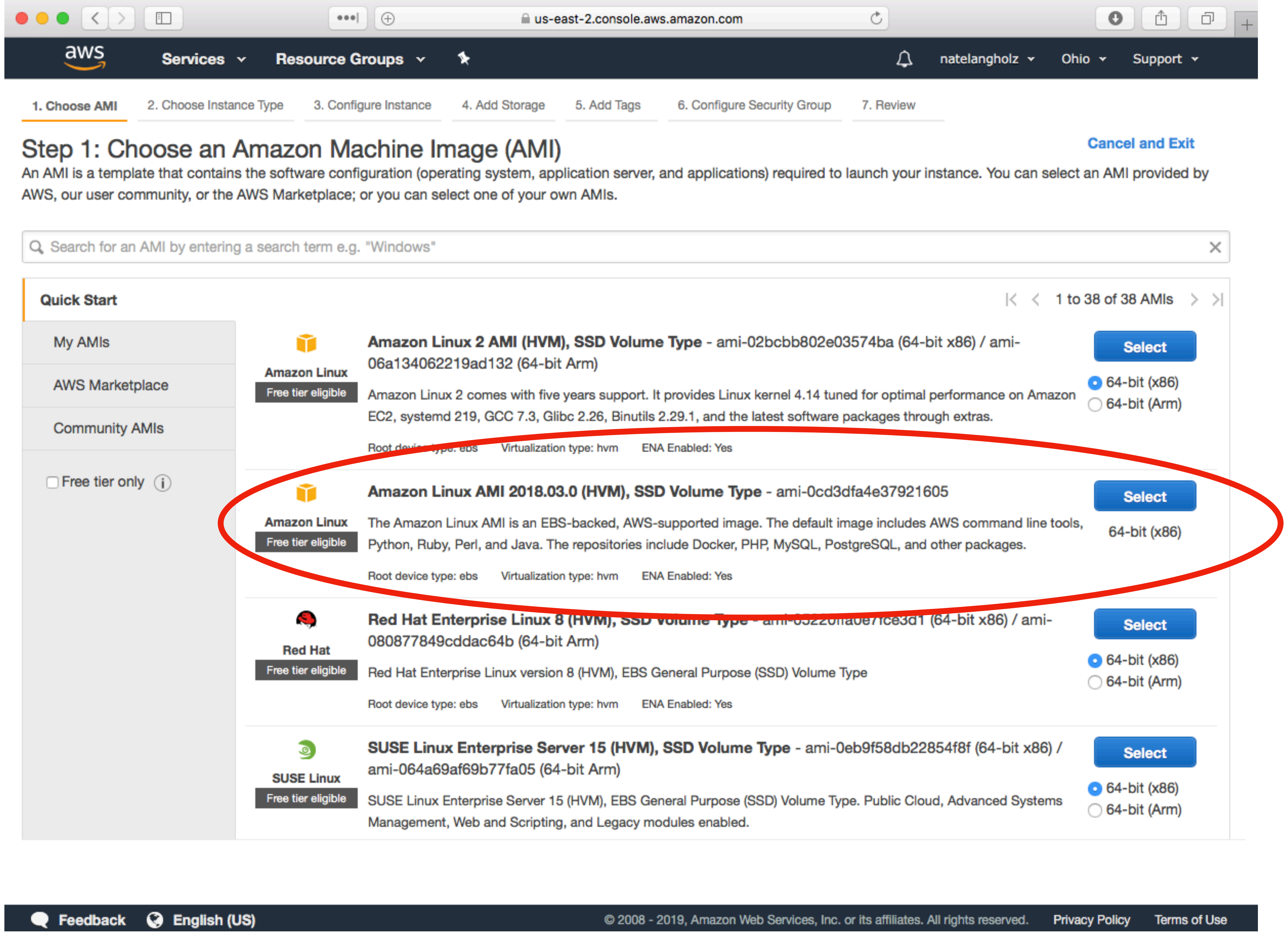
Barracuda CloudGen Firewall for AWS - PAYG

By Barracuda Networks, Inc.
Rating ★★★★★
Starting from $0.60/hr or from $4,599/yr (12% savings) for software + AWS usage fees
View all Infrastructure Software

Matillion ETL for Amazon Redshift

Feedback English (US) © 2008 - 2019, Amazon Web Services, Inc. or its affiliates. All rights reserved. Privacy Policy Terms of Use

Choose an Amazon Machine Instance from the options.



1. Choose AMI   2. Choose Instance Type   3. Configure Instance   4. Add Storage   5. Add Tags   6. Configure Security Group   7. Review

Cancel and Exit

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.

Search for an AMI by entering a search term e.g. "Windows"

Quick Start

1 to 38 of 38 AMIs

My AMIs
AWS Marketplace
Community AMIs

□ Free tier only

Amazon Linux 2 AMI (HVM), SSD Volume Type - ami-02bcbb802e03574ba (64-bit x86) / ami-06a134062219ad132 (64-bit Arm)

Amazon Linux · Free tier eligible

Amazon Linux 2 comes with five years support. It provides Linux kernel 4.14 tuned for optimal performance on Amazon EC2, systemd 219, GCC 7.3, Glibc 2.26, Binutils 2.29.1, and the latest software packages through extras.

Root device type: ebs   Virtualization type: hvm   ENA Enabled: Yes

Select   ● 64-bit (x86)   ○ 64-bit (Arm)

Amazon Linux AMI 2018.03.0 (HVM), SSD Volume Type - ami-0cd3dfa4e37921605

Amazon Linux · Free tier eligible

The Amazon Linux AMI is an EBS-backed, AWS-supported image. The default image includes AWS command line tools, Python, Ruby, Perl, and Java. The repositories include Docker, PHP, MySQL, PostgreSQL, and other packages.

Root device type: ebs   Virtualization type: hvm   ENA Enabled: Yes

Select   64-bit (x86)

Red Hat Enterprise Linux 8 (HVM), SSD Volume Type - ami-05220ffa0e71ce3d1 (64-bit x86) / ami-080877849cddac64b (64-bit Arm)

Red Hat · Free tier eligible

Red Hat Enterprise Linux version 8 (HVM), EBS General Purpose (SSD) Volume Type

Root device type: ebs   Virtualization type: hvm   ENA Enabled: Yes

Select   ● 64-bit (x86)   ○ 64-bit (Arm)

SUSE Linux Enterprise Server 15 (HVM), SSD Volume Type - ami-0eb9f58db22854f8f (64-bit x86) / ami-064a69af69b77fa05 (64-bit Arm)

SUSE Linux · Free tier eligible

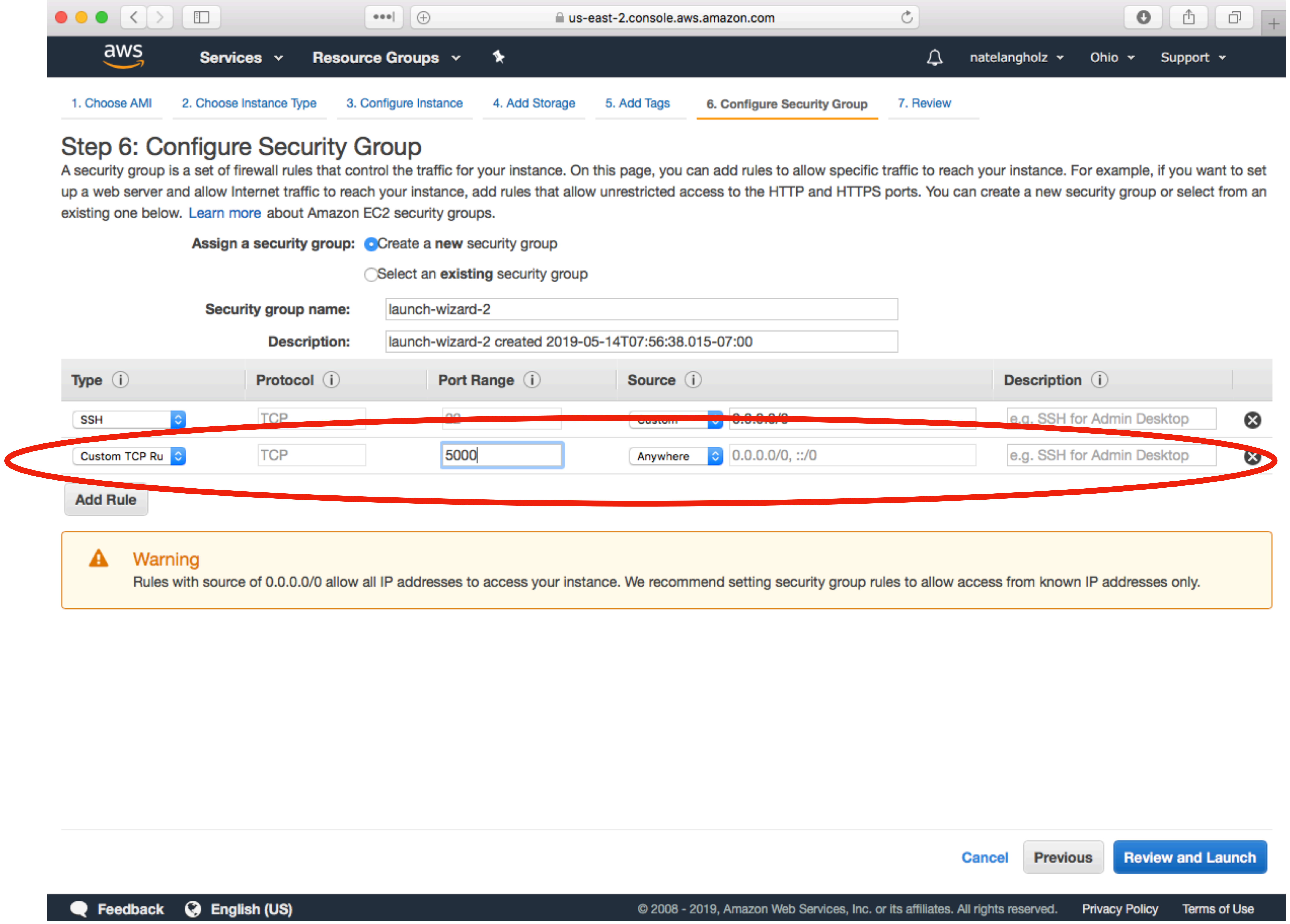
SUSE Linux Enterprise Server 15 (HVM), EBS General Purpose (SSD) Volume Type. Public Cloud, Advanced Systems Management, Web and Scripting, and Legacy modules enabled.

Select   ● 64-bit (x86)   ○ 64-bit (Arm)

Feedback   English (US)   © 2008 - 2019, Amazon Web Services, Inc. or its affiliates. All rights reserved.   Privacy Policy   Terms of Use

Using the Amazon Linux AMI because its Free tier eligible and the repos include Docker. (I believe the Linux 2 AMI would also work)

Configure Security Group; click `add rule`; the new rule will allow inbound requests



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

Security group name: launch-wizard-2

Description: launch-wizard-2 created 2019-05-14T07:56:38.015-07:00

Type	Protocol	Port Range	Source		Description
SSH	TCP	22	Custom	0.0.0.0/0	e.g. SSH for Admin Desktop
Custom TCP Ru	TCP	5000	Anywhere	0.0.0.0/0, ::/0	e.g. SSH for Admin Desktop

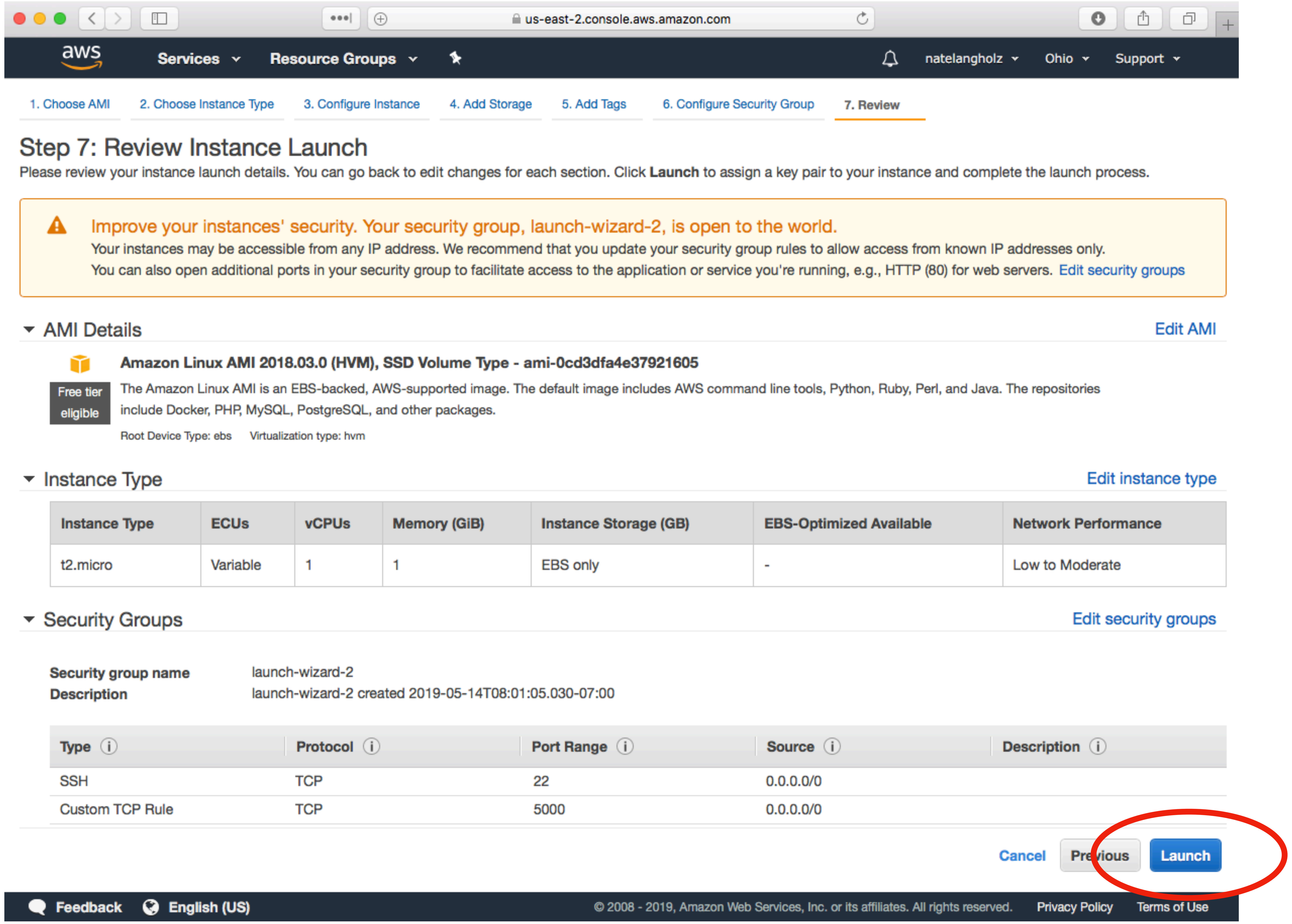
Warning
Rules with source of 0.0.0.0/0 allow all IP addresses to access your instance. We recommend setting security group rules to allow access from known IP addresses only.

Cancel
Previous
Review and Launch
Feedback
English (US)
© 2008 - 2019, Amazon Web Services, Inc. or its affiliates. All rights reserved.
Privacy Policy
Terms of Use

Pick `Custom TCP Rule`
Enter 5000 into the Port Range (match whatever port you have used in your local app)
Change Source to `Anywhere`

Review and Launch

You'll notice the warning about improving security. For what we're doing this should be okay not to worry about.



Step 7: Review Instance Launch

Please review your instance launch details. You can go back to edit changes for each section. Click **Launch** to assign a key pair to your instance and complete the launch process.

Improve your instances' security. Your security group, launch-wizard-2, is open to the world.

Your instances may be accessible from any IP address. We recommend that you update your security group rules to allow access from known IP addresses only.

You can also open additional ports in your security group to facilitate access to the application or service you're running, e.g., HTTP (80) for web servers. Edit security groups

AMI Details

Edit AMI

Amazon Linux AMI 2018.03.0 (HVM), SSD Volume Type - ami-0cd3dfa4e37921605

Free tier eligible

The Amazon Linux AMI is an EBS-backed, AWS-supported image. The default image includes AWS command line tools, Python, Ruby, Perl, and Java. The repositories include Docker, PHP, MySQL, PostgreSQL, and other packages.

Root Device Type: ebs Virtualization type: hvm

Instance Type

Edit instance type

Instance Type	ECUs	vCPUs	Memory (GiB)	Instance Storage (GB)	EBS-Optimized Available	Network Performance
t2.micro	Variable	1	1	EBS only	-	Low to Moderate

Security Groups

Edit security groups

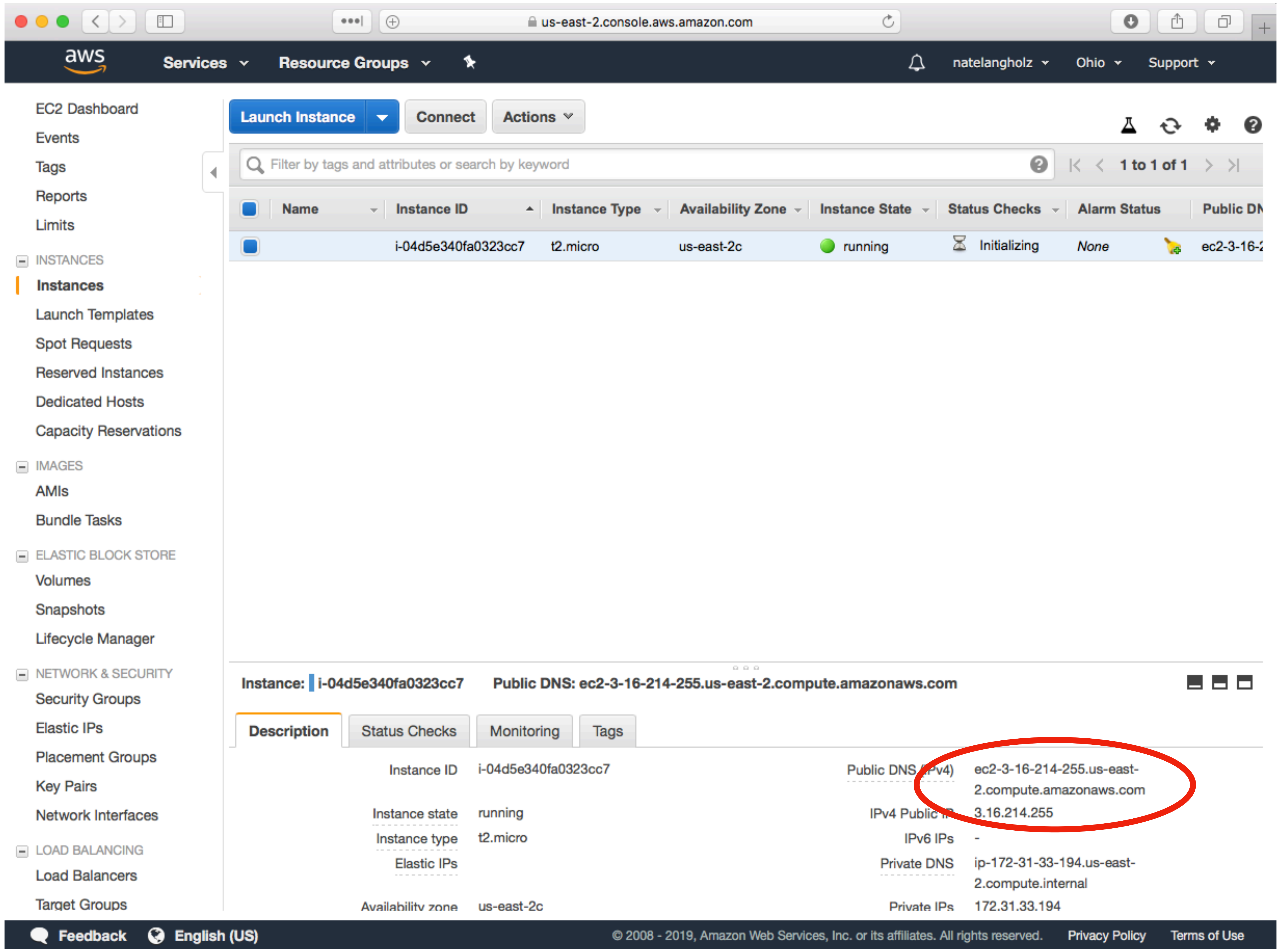
Security group name	launch-wizard-2
Description	launch-wizard-2 created 2019-05-14T08:01:05.030-07:00

Type	Protocol	Port Range	Source	Description
SSH	TCP	22	0.0.0.0/0	
Custom TCP Rule	TCP	5000	0.0.0.0/0	

Cancel
Previous
Launch
Feedback
English (US)
© 2008 - 2019, Amazon Web Services, Inc. or its affiliates. All rights reserved.
Privacy Policy
Terms of Use

Should be okay to Launch at this point

ec2 Dashboard



Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks	Alarm Status	Public DN
	i-04d5e340fa0323cc7	t2.micro	us-east-2c	● running	Initializing	None	ec2-3-16-2

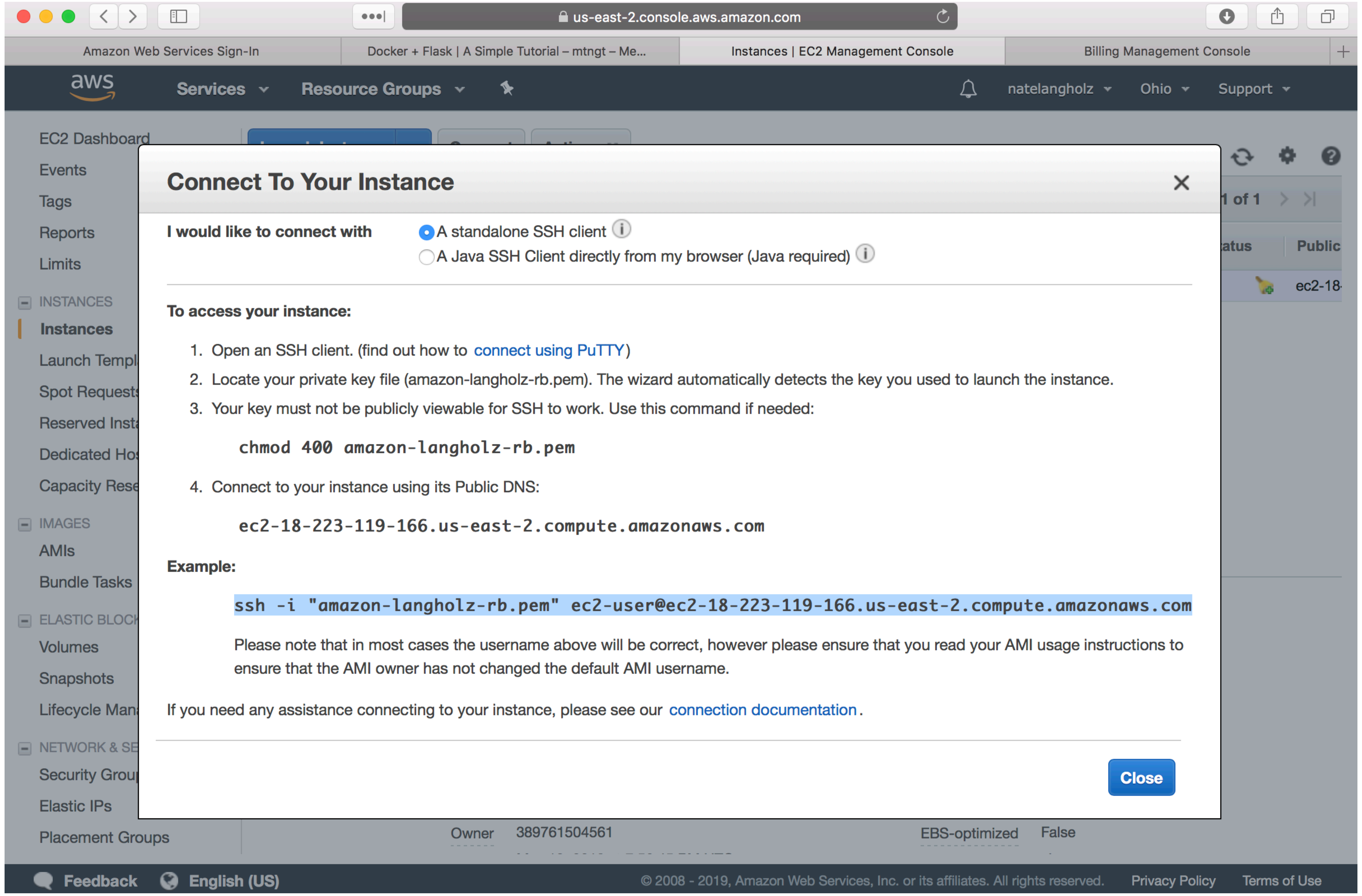
Instance: i-04d5e340fa0323cc7 Public DNS: ec2-3-16-214-255.us-east-2.compute.amazonaws.com

Instance ID	i-04d5e340fa0323cc7	Public DNS (IPv4)	ec2-3-16-214-255.us-east-2.compute.amazonaws.com
Instance state	running	IPv4 Public IP	3.16.214.255
Instance type	t2.micro	IPv6 IPs	-
Elastic IPs		Private DNS	ip-172-31-33-194.us-east-2.compute.internal
Availability zone	us-east-2c	Private IPs	172.31.33.194

Observe that your Instance state is running

Also note your Public DNS, as this is where you will access this instance

Connect to your instance; this is an overview of the instructions. I’ll try to detail them a bit more. (See these by clicking `Connect` in your dashboard)



Connect To Your Instance

I would like to connect with

● A standalone SSH client

○ A Java SSH Client directly from my browser (Java required)

To access your instance:

1. Open an SSH client. (find out how to connect using PuTTY)
2. Locate your private key file (amazon-langholz-rb.pem). The wizard automatically detects the key you used to launch the instance.
3. Your key must not be publicly viewable for SSH to work. Use this command if needed:

```
chmod 400 amazon-langholz-rb.pem
```

4. Connect to your instance using its Public DNS:

```
ec2-18-223-119-166.us-east-2.compute.amazonaws.com
```

Example:

```
ssh -i "amazon-langholz-rb.pem" ec2-user@ec2-18-223-119-166.us-east-2.compute.amazonaws.com
```

Please note that in most cases the username above will be correct, however please ensure that you read your AMI usage instructions to ensure that the AMI owner has not changed the default AMI username.

If you need any assistance connecting to your instance, please see our connection documentation.

Close

Let’s ssh into our AMI

key-file.pem

ec2-user@public-dns

```
ssh -i "amazon-langholz-rb.pem" ec2-user@ec2-18-223-119-166.us-east-2.compute.amazonaws.com
```

Will need to be in the directory of my pem key

Time to install Docker and Docker-compose

Update the environment installed packages and cache

```
sudo yum update –y
```

Install Docker

```
sudo yum install –y docker
```

Add the ec2-user to the docker group so you can execute Docker commands without sudo

```
sudo usermod –a –G docker ec2–user
```

this pulls docker-compose from GitHub (the first should be all one line! Thats why I’ve made it so small here)

```
sudo curl -L https://github.com/docker/compose/releases/download/1.21.0/docker-compose-`uname -s`-`uname -m` | sudo tee /usr/local/bin/docker-compose > /dev/null
```

Then

```
sudo chmod +x /usr/local/bin/docker-compose
```

Start the docker service

```
sudo service docker start
sudo chkconfig docker on
```

Check docker-compose version

```
docker–compose ––version
```

We can get a zip file through (or scp files…really we only need the docker dir)

```
wget https://github.com/natelangholz/shiny-app-418/archive/master.zip
```

Unzip using

```
unzip master.zip
```

And then remove the zip file as we no longer need it

```
rm master.zip
```

Docker on AMI

Same as locker

Change your directory into the docker directory.

Then (remember include -d if you want it detached an your prompt back):

```
docker-compose up
```

Now check http://<your-public-dns>:3838

To stop your API, within your AMI stop your docker container as usual.

Either

```
Docker–compose stop
```

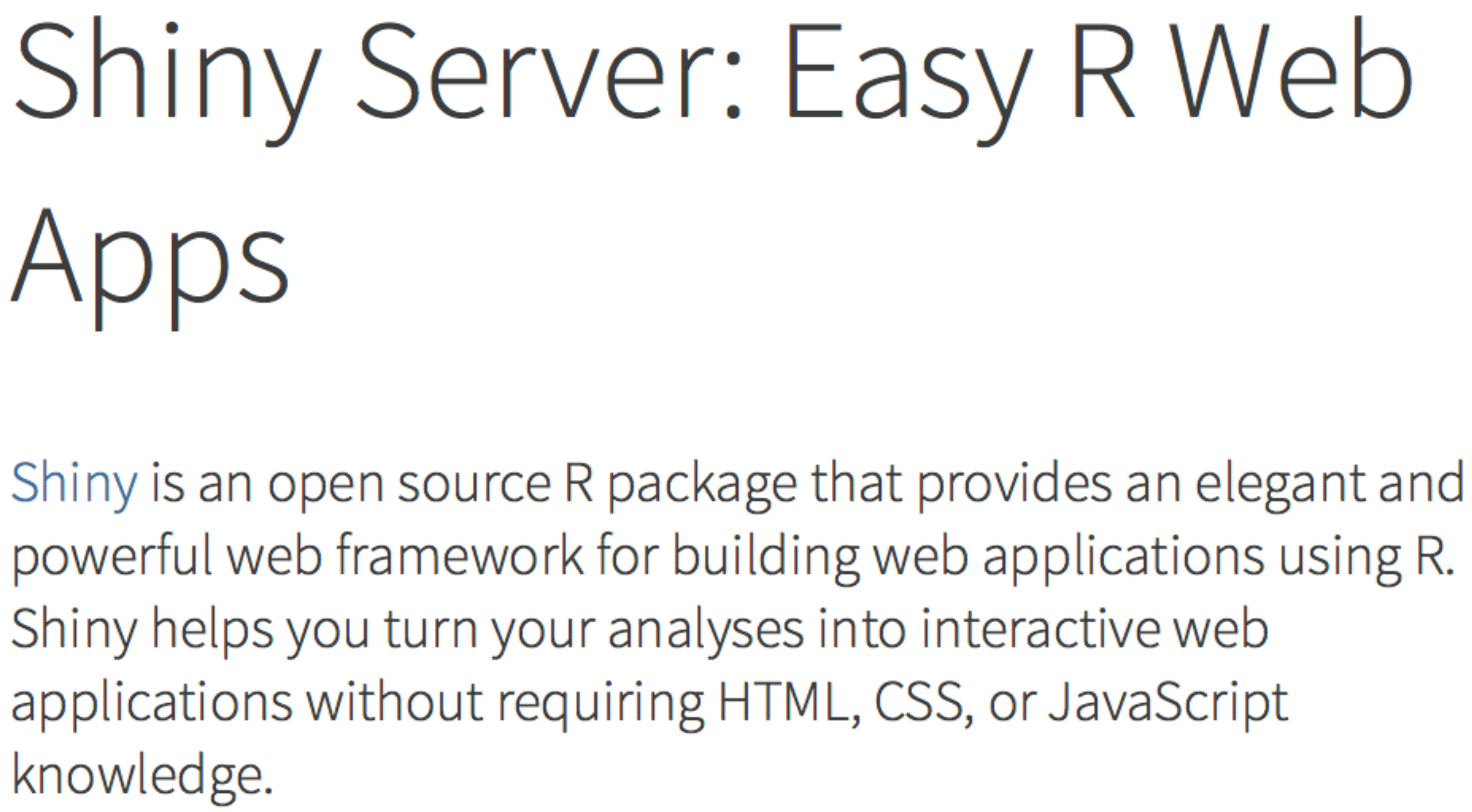
or

```
Docker container ls

Docker container kill <container–name>
```

Then exit your AMI by simply typing `exit`

What is a Shiny App?



Shiny Server: Easy R Web Apps

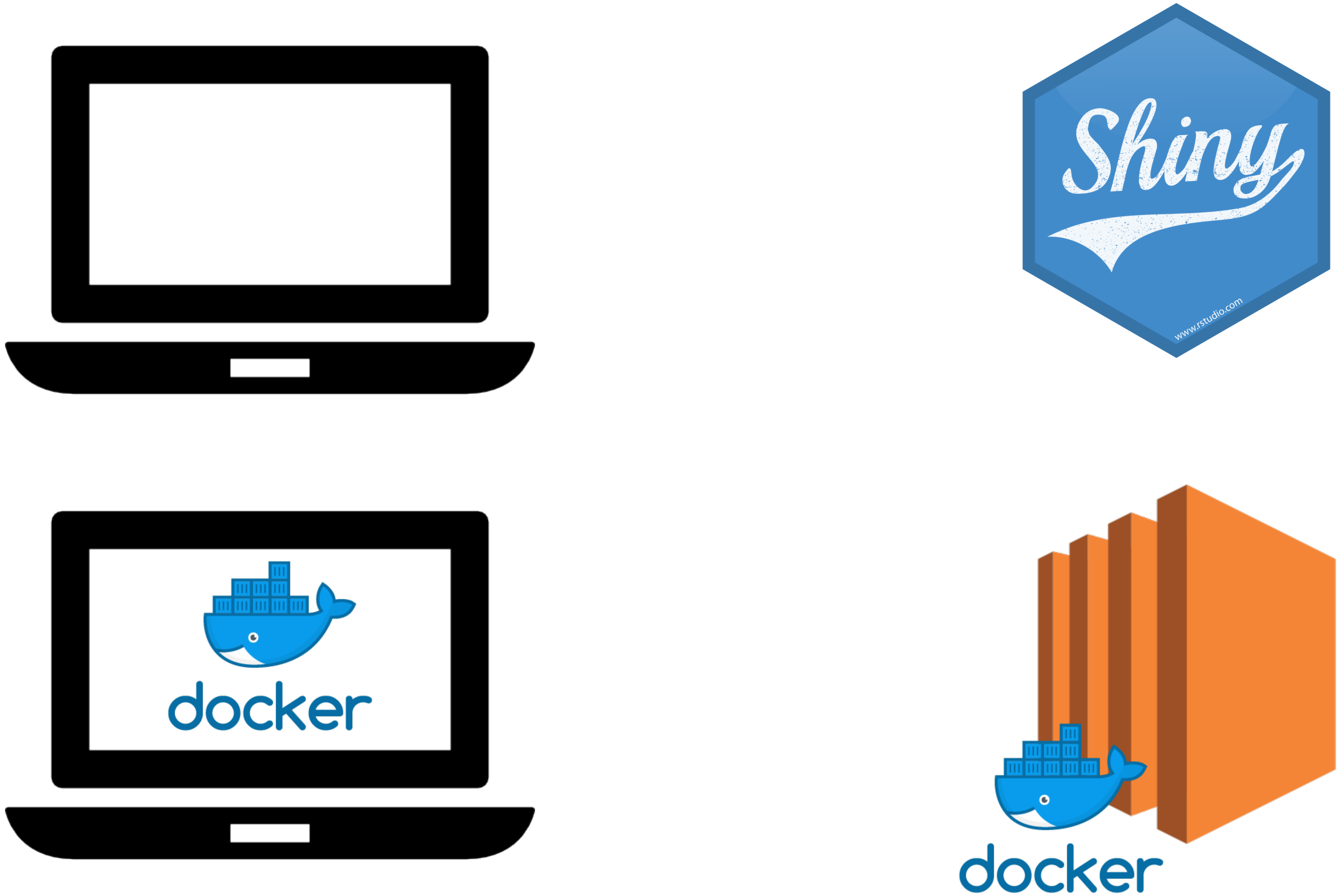
Shiny is an open source R package that provides an elegant and powerful web framework for building web applications using R. Shiny helps you turn your analyses into interactive web applications without requiring HTML, CSS, or JavaScript knowledge.

How to build one?

Honestly, look at Rstudio’s tutorials

https://shiny.rstudio.com/tutorial/

Deploying a Shiny App 4 Different Ways!



Locally, in terminal

Git clone or zip download my plumber-api-repo (not in your forked course repo!)

```
R –e "shiny::runApp('app/')"
```

Docker on Local Machine

Git clone or zip download my plumber-api-repo (not in your forked course repo!)

Change your directory into the docker folder.

Then (remember include -d if you want it detached an your prompt back):

```
docker–compose up
```

Now check http://localhost:3838

Okay, so far we have done that locally for both

Now let’s look at deploying it on the cloud so anyone can look at it

Let’s start with an AWS ec2 instance. I would recommend creating a new instance to deploy this.

We can use the same .pem key (will probably need to move it but that’s fine)

We can get a zip file through (or scp files…really we only need the docker dir)

```
wget https://github.com/natelangholz/shiny-app-418/archive/master.zip
```

Unzip using

```
unzip master.zip
```

And then remove the zip file as we no longer need it

```
rm master.zip
```

Docker on AMI

Same as locker

Change your directory into the docker directory.

Then (remember include -d if you want it detached an your prompt back):

```
docker–compose up
```

Now check http://<your-public-dns>:3838

To stop your API, within your AMI stop your docker container as usual.

Either

```
Docker–compose stop
```

or

```
Docker container ls

Docker container kill <container–name>
```

Then exit your AMI by simply typing `exit`

Deploy to shinyapps.io

You will need to create an account here. Do so now.

You will need your token and your secret key (don’t push to Github!)

This is probably easier to deploy something than AWS but at the free level you didn’t get as much runtime so you will have to pay sooner (if you are getting lots of traffic)

Deploy to shinyapps.io

You will need to create an account here. Do so now.

You will need your token and your secret key (don’t push to Github!)

This is probably easier to deploy something than AWS but at the free level you didn’t get as much runtime so you will have to pay sooner (if you are getting lots of traffic)