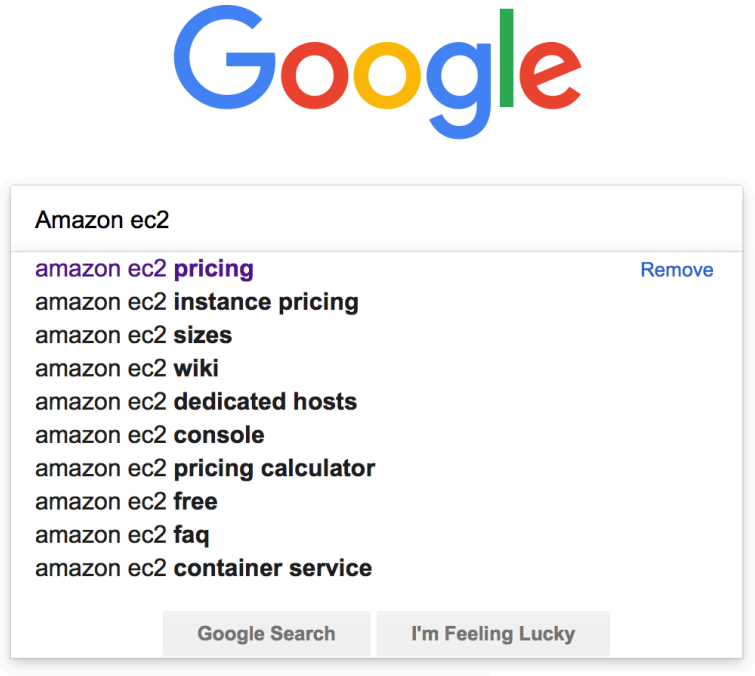
A GUIDE TO USING AMAZON-AWS TO CONDUCT PARALLEL PROCESSING IN R.

PERRY WILLIAMS

Navigate to the Amazon EC2 website



Select Amazon EC2 – Amazon AWS:

About 23,700,000 results (0.72 seconds)

Amazon Elastic Compute Cloud | Virtual Server Hosting
Ad aws.amazon.com/ec2
Complete Control of Your Servers. Scale Capacity in Minutes. Get Started Today!
Multiple Instance Types · Easy To Start · Quickly Scale Capacity · Virtual Private Cloud
Types: Elastic Block Store, EBS-Optimized Instances, Virtual Private Cloud, Elastic Load Balancing

- **Instance Types** — Our Instance Types are Optimized to Fit Your Use Case. T2, M4 & More.
- **Pricing** — Pay For What You Use. No Minimum Fee. Estimate Your Monthly Bill Now
- **Get Started For Free** — Free Usage of 40+ AWS Products. EC2, S3, DynamoDB & More.
- **Product Details** — Elastic IP Addresses, Auto Scaling, HPC Clusters, & More.

Amazon EC2 - Amazon AWS
https://aws.amazon.com/ec2/
Amazon EC2's simple web service interface allows you to obtain and configure capacity with minimal friction. It provides you with complete control of your computing resources and lets you run on Amazon's proven computing environment. **Amazon EC2** reduces the time required to obtain and boot new server instances to ...

- **Amazon EC2 Pricing** — Learn about the four ways to pay for Amazon EC2 instances: On ...
- **Amazon EC2 Instance Types** — Amazon EC2 provides a wide selection of instance types ...
- **Amazon EC2 Product Details** — Amazon EC2 presents a true virtual computing environment ...
- **Amazon EC2 for Windows ...** — The AWS Free Usage Tier includes Amazon EC2 ...
- **Amazon Elastic Compute ...** — Amazon Elastic Compute Cloud (Amazon EC2) is a web service ...
- **Amazon EC2 Spot Instances** — Amazon EC2 Spot instances are spare compute capacity in the ...

More results from amazon.com »

What Is Amazon EC2? - Amazon Elastic Compute Cloud
https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/concepts.html
Amazon Elastic Compute Cloud (**Amazon EC2**) provides scalable computing capacity in the **Amazon** Web Services (**AWS**) cloud. ... Persistent storage volumes for your data using **Amazon** Elastic Block Store (**Amazon** EBS), known as **Amazon** EBS volumes. Multiple physical locations for your resources ...
Features of Amazon EC2 · How to Get Started with ... · Accessing Amazon EC2

Amazon Elastic Compute Cloud
Computer application

Amazon Elastic Compute Cloud forms a central part of Amazon.com's cloud-computing platform, Amazon Web Services, by allowing users to rent virtual computers on which to run their own computer applications. Wikipedia

Developed by: Amazon.com
Original author(s): Amazon.com, Inc
Initial release: August 25, 2006; 11 years ago (public beta)
Operating system: Linux; Microsoft Windows; FreeBSD

People also search for View 15+

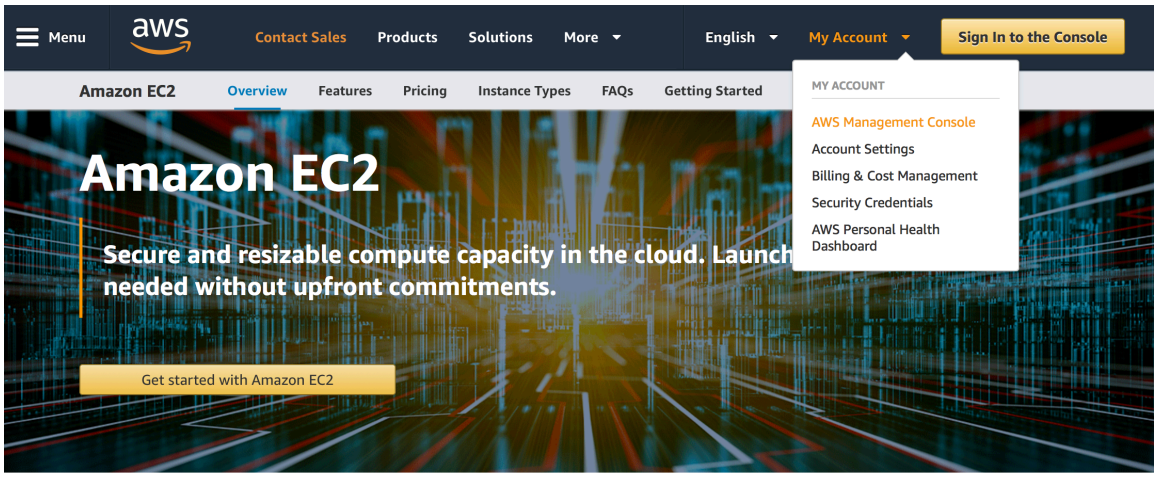

Linux


Microsoft Windows

Microsoft Azure

Feedback

Select “My Account” and “AWS Management Console.” If you haven’t created an account you may need to do so.



Amazon Elastic Compute Cloud (Amazon EC2) is a web service that provides secure, resizable compute capacity in the cloud. It is designed to make web-scale cloud computing easier for developers.

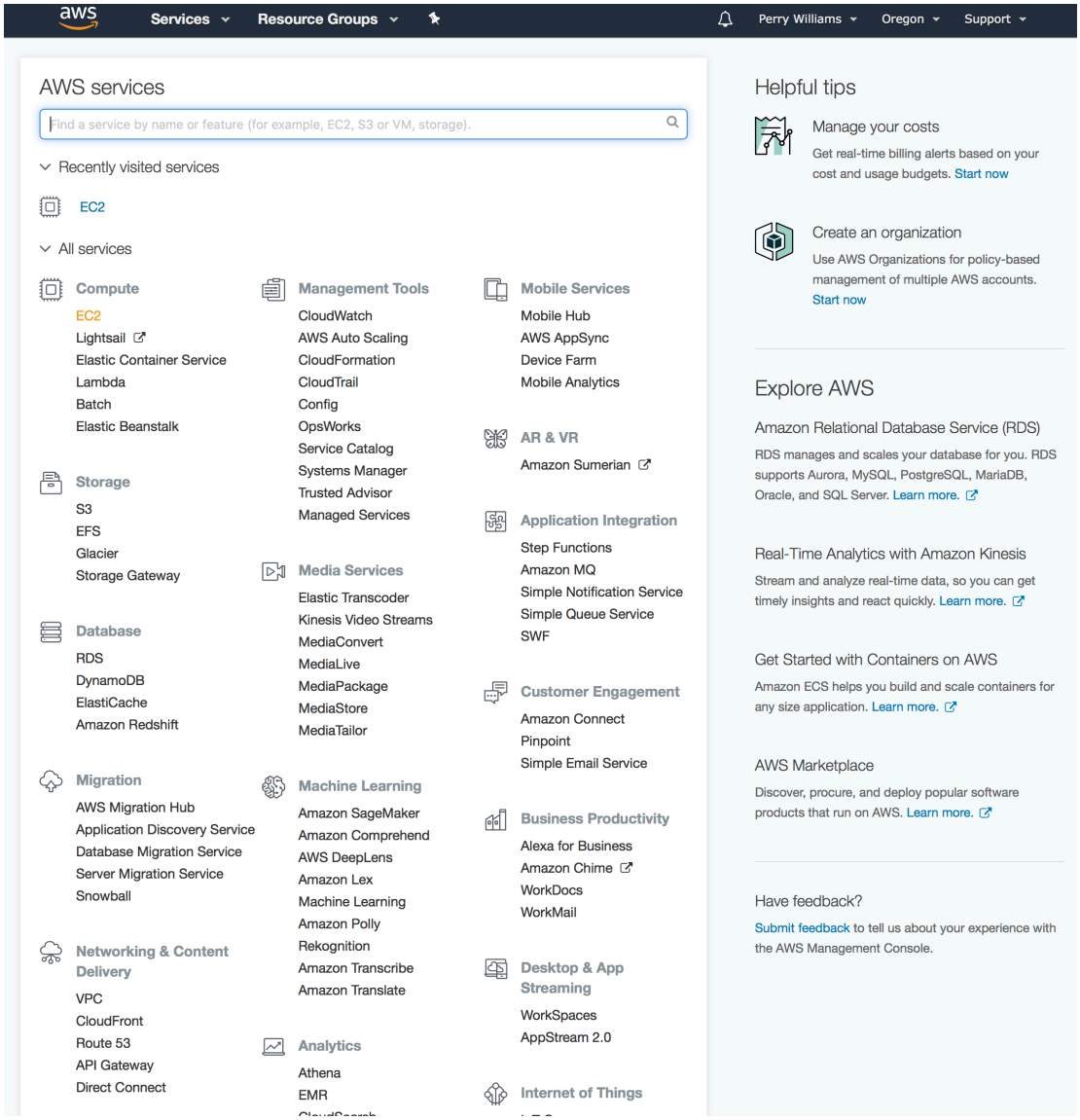
Amazon EC2’s simple web service interface allows you to obtain and configure capacity with minimal friction. It provides you with complete control of your computing resources and lets you run on Amazon’s proven computing environment. Amazon EC2 reduces the time required to obtain and boot new server instances to minutes, allowing you to quickly scale capacity, both up and down, as your computing requirements change. Amazon EC2 changes the economics of computing by allowing you to pay only for capacity that you actually use. Amazon EC2 provides developers the tools to build failure resilient applications and isolate them from common failure scenarios.

Try Amazon EC2 for Free

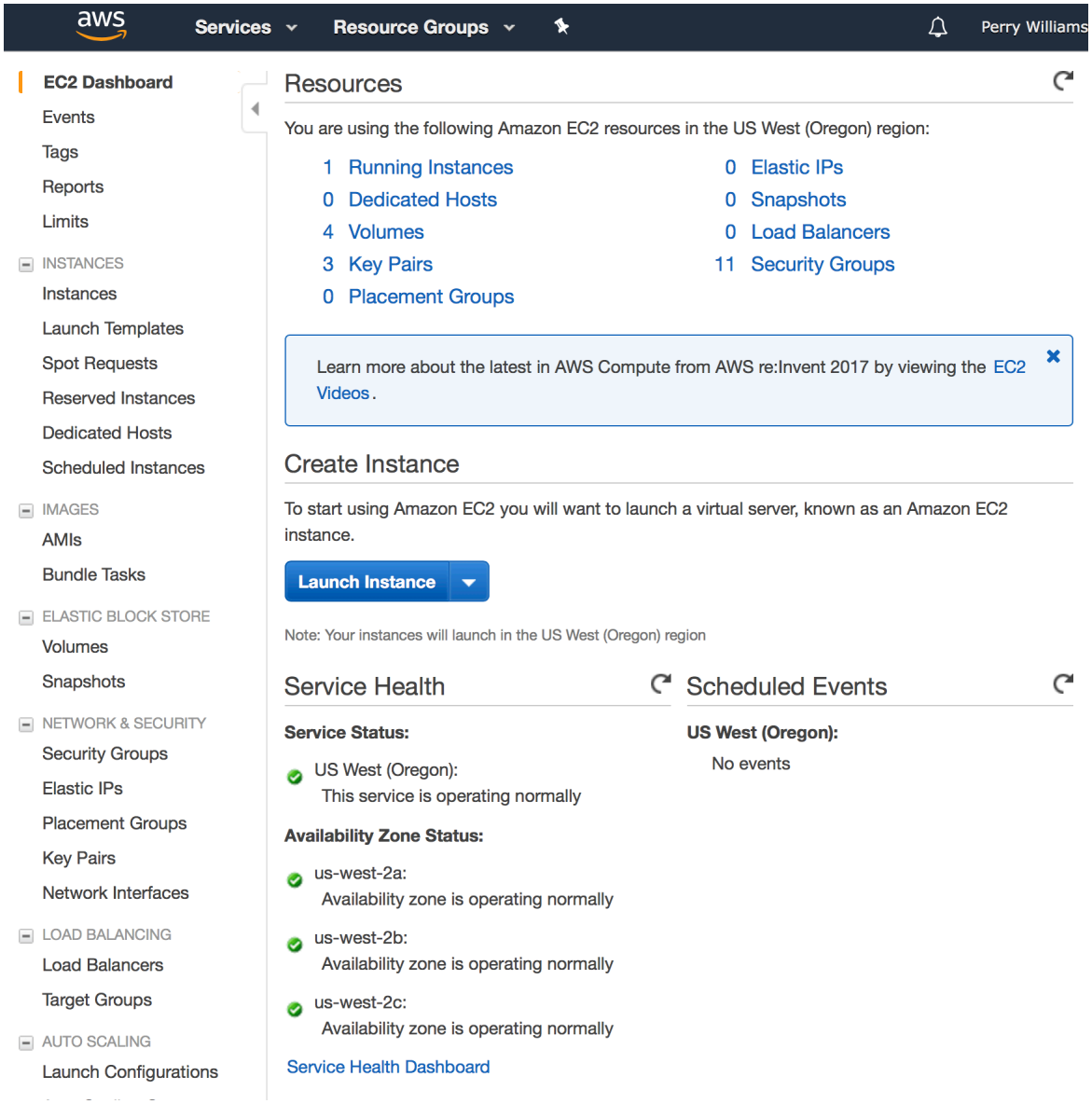
AWS Free Tier includes 750 hours of Linux and Windows t2.micro instances each month for one year. To stay within the Free Tier, use only EC2 Micro instances.

View AWS Free Tier details >>

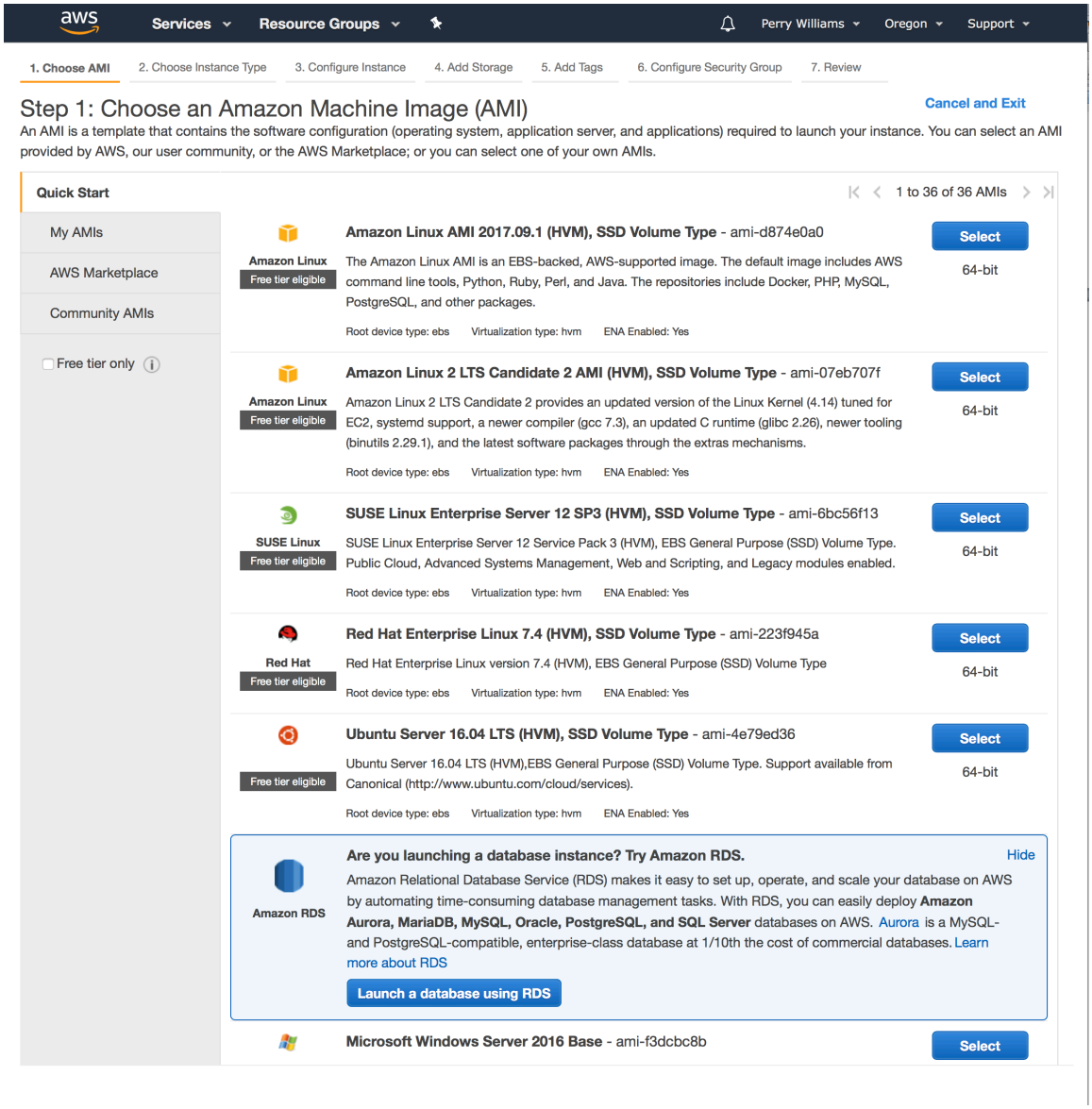
Select EC2:


aws
Services
Resource Groups
Perry Williams
Oregon
Support
AWS services
Find a service by name or feature (for example, EC2, S3 or VM, storage).
Recently visited services
EC2
All services
Compute
EC2
Lightsail
Elastic Container Service
Lambda
Batch
Elastic Beanstalk
Storage
S3
EFS
Glacier
Storage Gateway
Database
RDS
DynamoDB
ElastiCache
Amazon Redshift
Migration
AWS Migration Hub
Application Discovery Service
Database Migration Service
Server Migration Service
Snowball
Networking & Content Delivery
VPC
CloudFront
Route 53
API Gateway
Direct Connect
Management Tools
CloudWatch
AWS Auto Scaling
CloudFormation
CloudTrail
Config
OpsWorks
Service Catalog
Systems Manager
Trusted Advisor
Managed Services
Media Services
Elastic Transcoder
Kinesis Video Streams
MediaConvert
MediaLive
MediaPackage
MediaStore
MediaTailor
Machine Learning
Amazon SageMaker
Amazon Comprehend
AWS DeepLens
Amazon Lex
Machine Learning
Amazon Polly
Rekognition
Amazon Transcribe
Amazon Translate
Analytics
Athena
EMR
Mobile Services
Mobile Hub
AWS AppSync
Device Farm
Mobile Analytics
AR & VR
Amazon Sumerian
Application Integration
Step Functions
Amazon MQ
Simple Notification Service
Simple Queue Service
SWF
Customer Engagement
Amazon Connect
Pinpoint
Simple Email Service
Business Productivity
Alexa for Business
Amazon Chime
WorkDocs
WorkMail
Desktop & App Streaming
WorkSpaces
AppStream 2.0
Internet of Things
Helpful tips
Manage your costs
Get real-time billing alerts based on your cost and usage budgets. Start now
Create an organization
Use AWS Organizations for policy-based management of multiple AWS accounts. Start now
Explore AWS
Amazon Relational Database Service (RDS)
RDS manages and scales your database for you. RDS supports Aurora, MySQL, PostgreSQL, MariaDB, Oracle, and SQL Server. Learn more.
Real-Time Analytics with Amazon Kinesis
Stream and analyze real-time data, so you can get timely insights and react quickly. Learn more.
Get Started with Containers on AWS
Amazon ECS helps you build and scale containers for any size application. Learn more.
AWS Marketplace
Discover, procure, and deploy popular software products that run on AWS. Learn more.
Have feedback?
Submit feedback to tell us about your experience with the AWS Management Console.

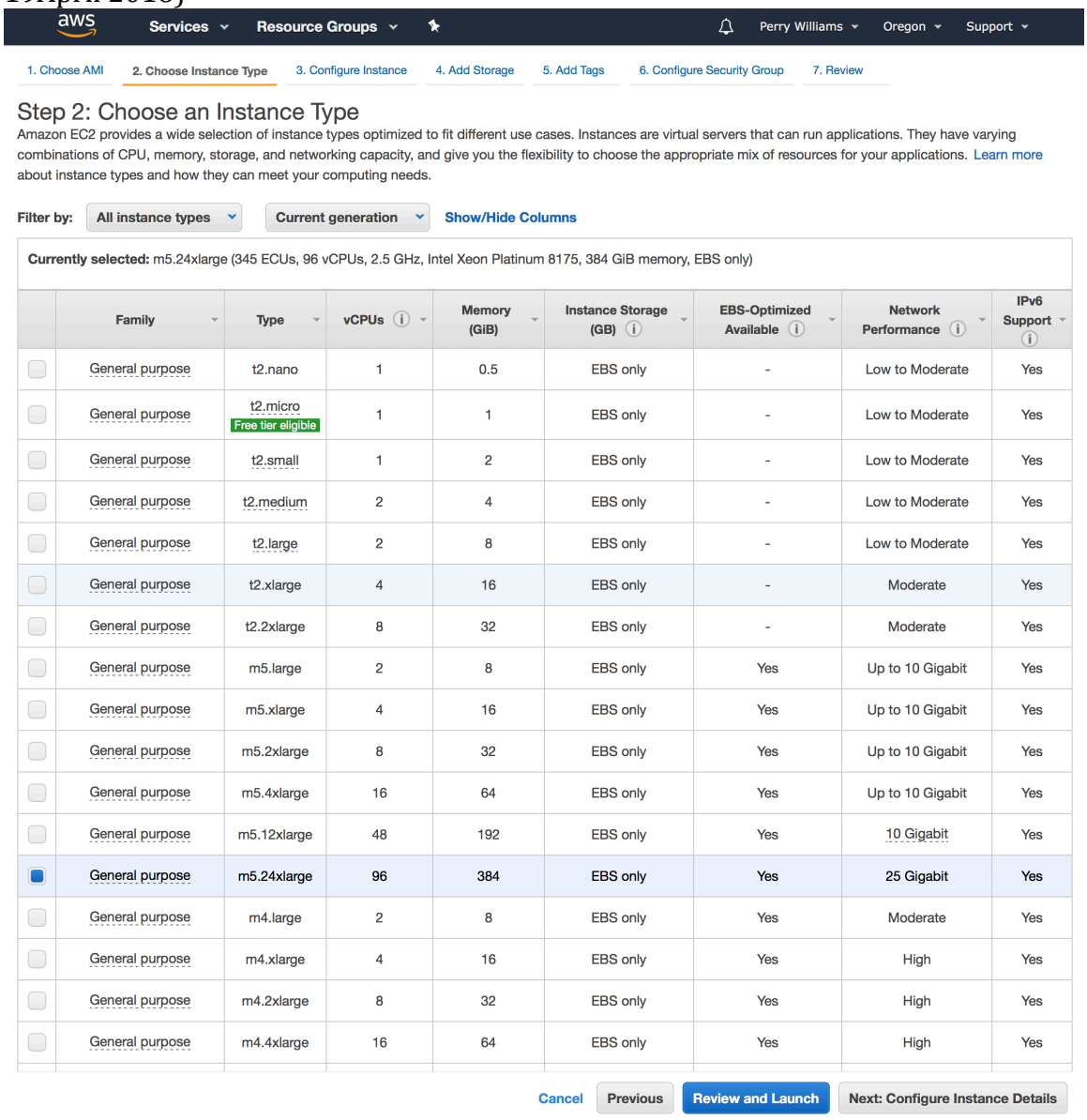
Launch Instance:



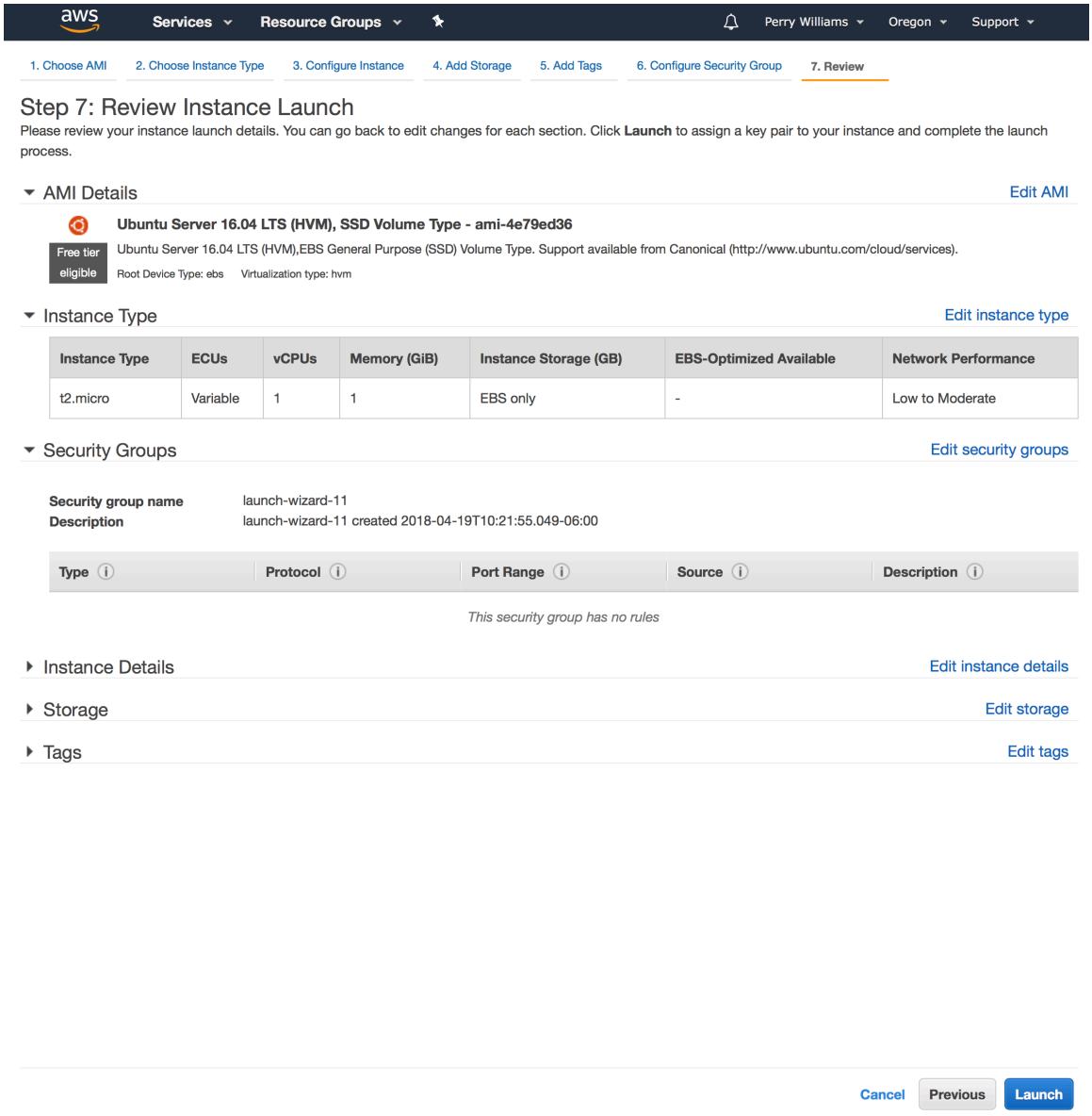
Using a Mac, I select Ubuntu Server 16.04 LTS (HVM), SSD Volume Type.



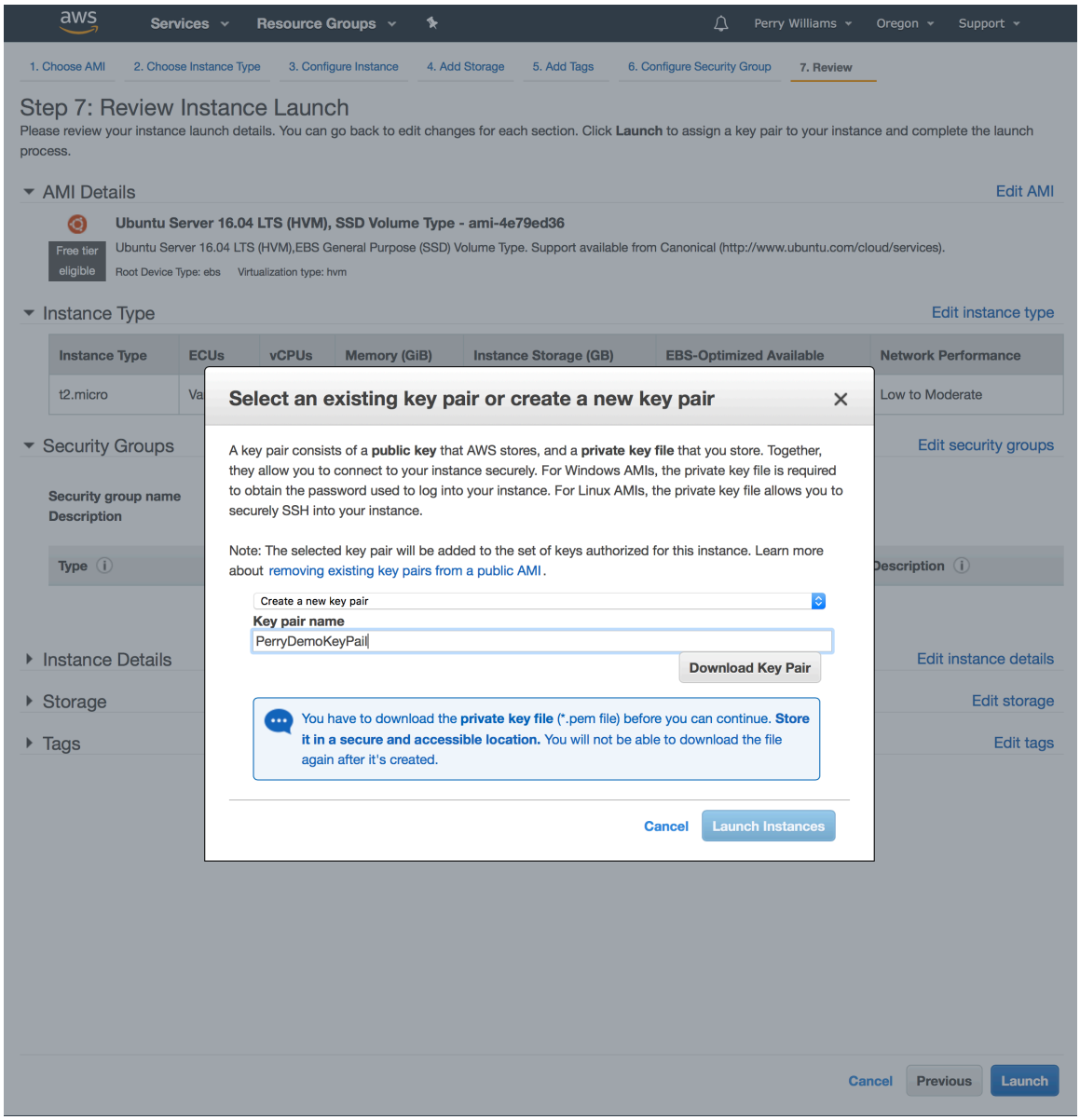
The m5.24xlarge Type has 96 cores that can be used. It costs: $4.608 per hour (as of 19April 2018)



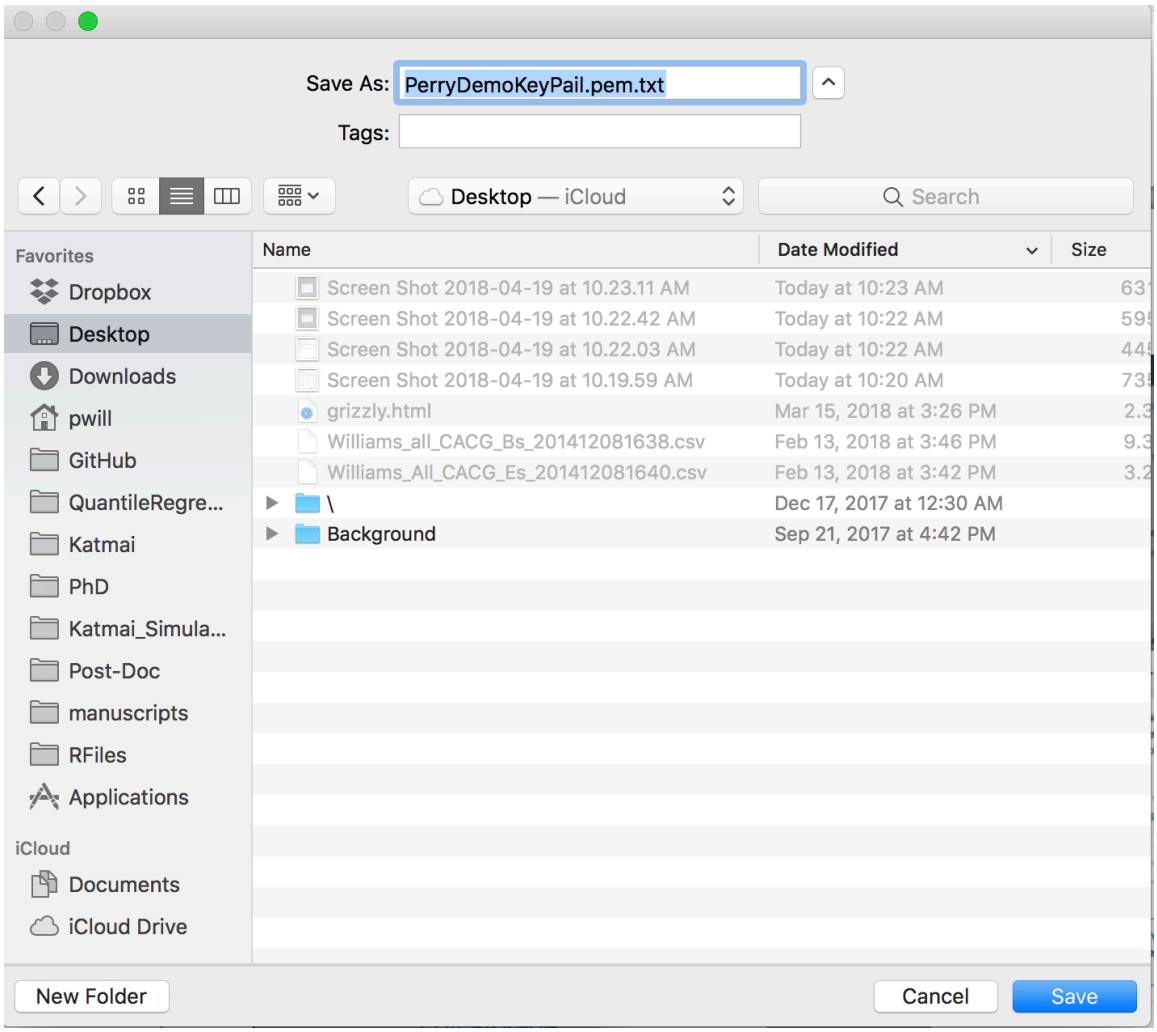
Select “Launch”



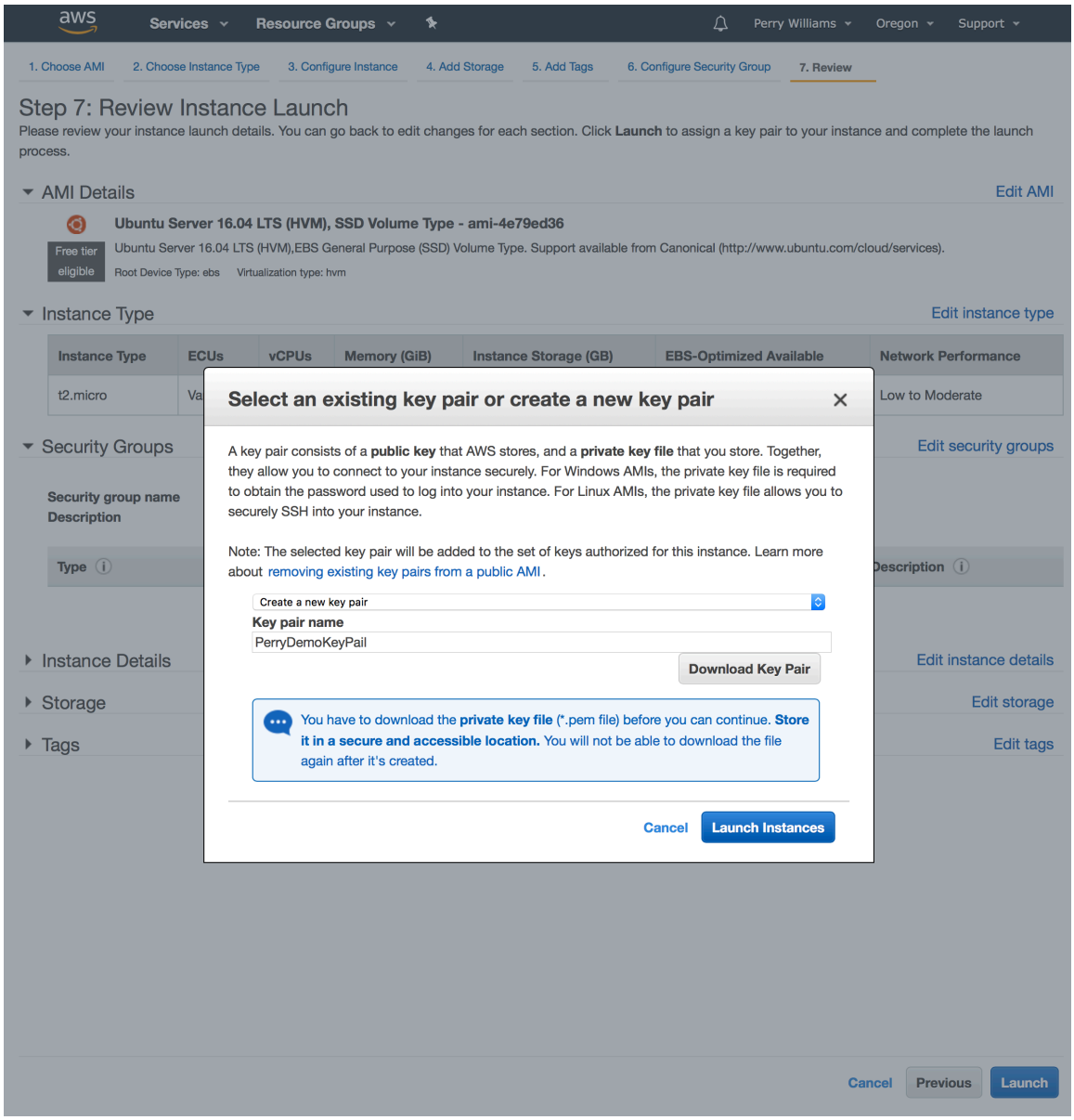
Create a KeyPair for security. When I originally created this document the keypair I created was called “PerryDemKeyPail”. I have since updated the keypair to “PerryKeyPailApril2018”, and use these interchangeably throughout this document.



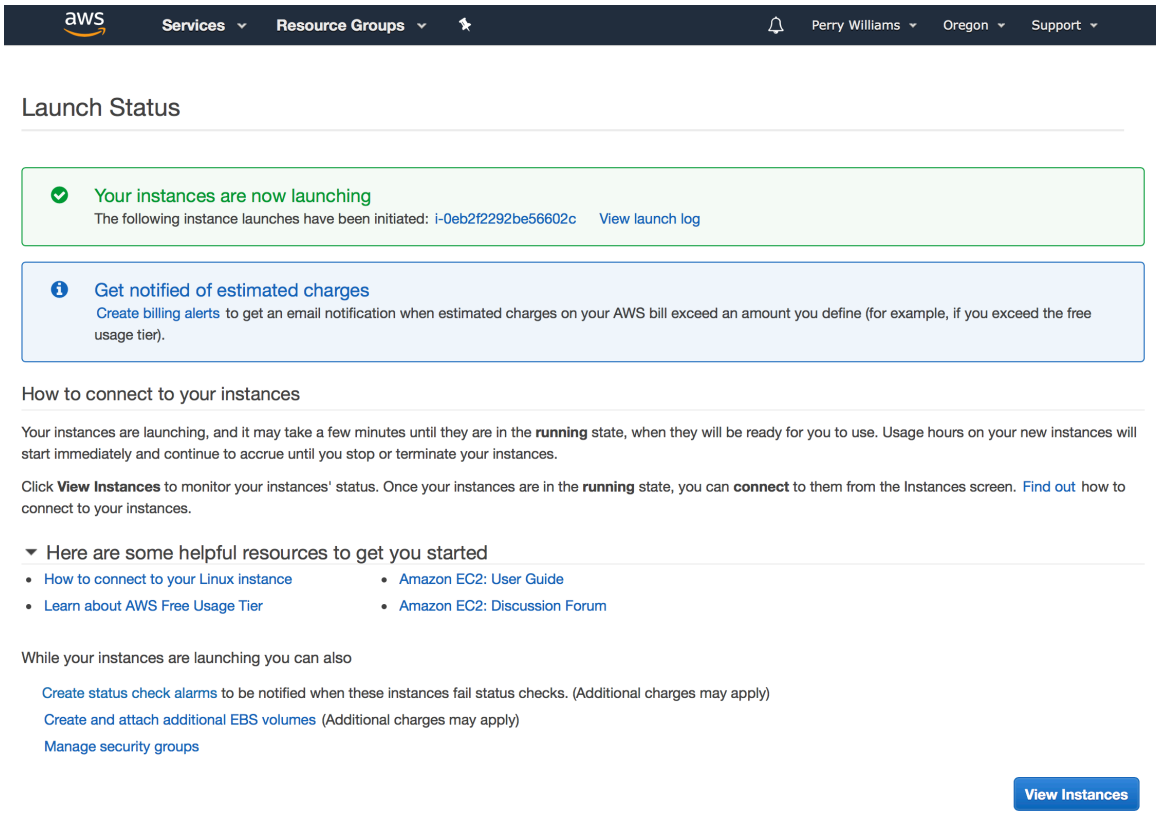
Save key pair in a place you'll be able to find it, but delete the ".txt" part listed below



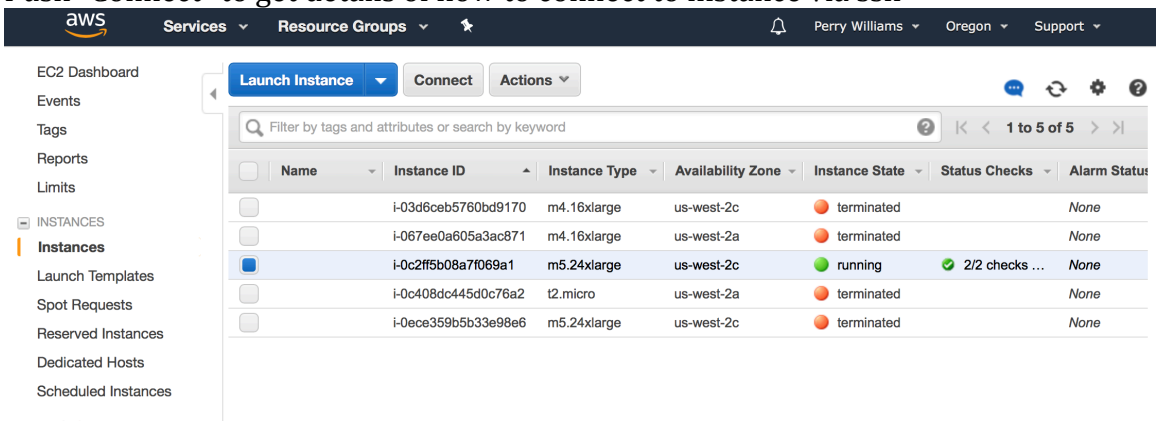
Select “Launch Instances”

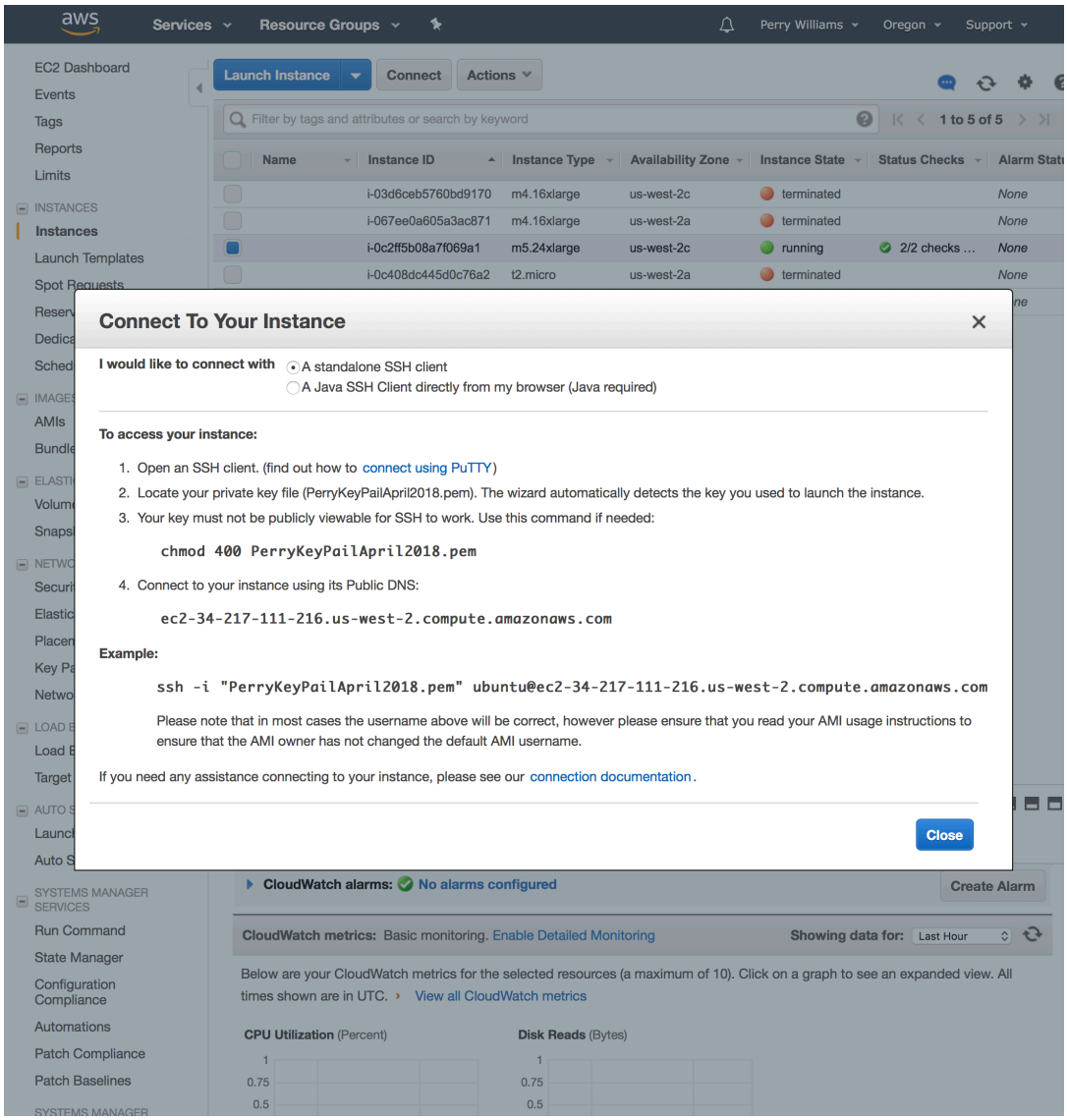


Select “View Instances”



Push “Connect” to get details of how to connect to instance via ssh

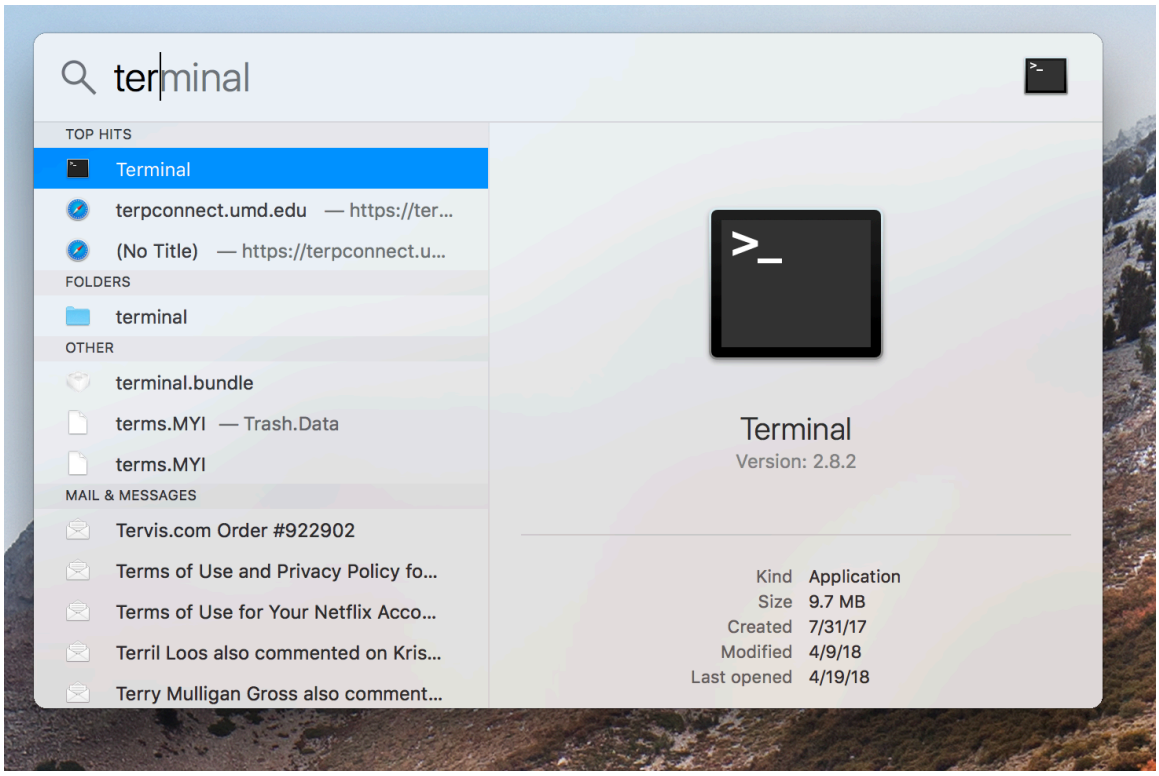




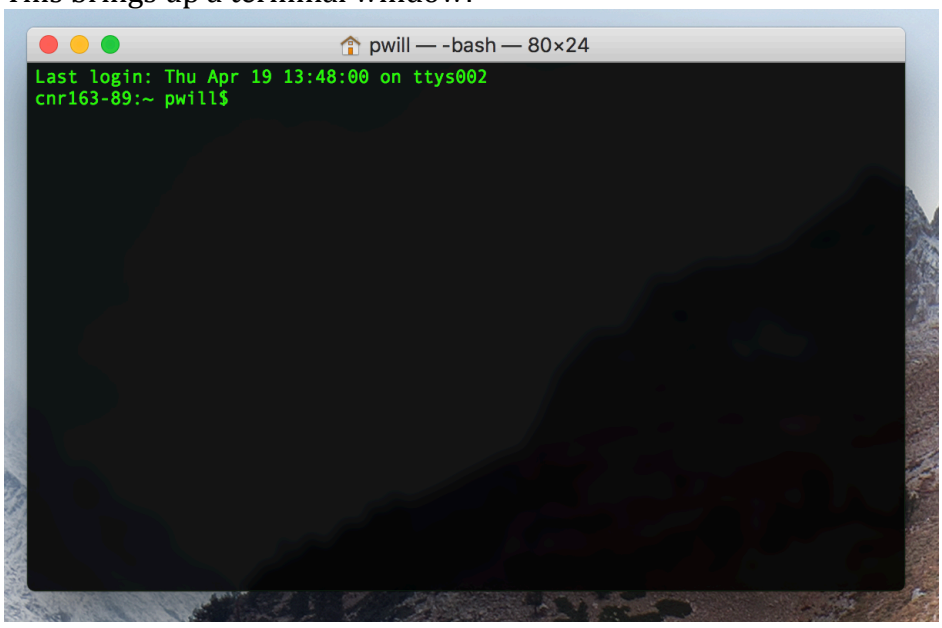
Follow the 4 steps to connect to instance:

1. Open a new terminal/ssh client.

On a Mac, Command>Space to open Finder, then “ter” followed by Return.

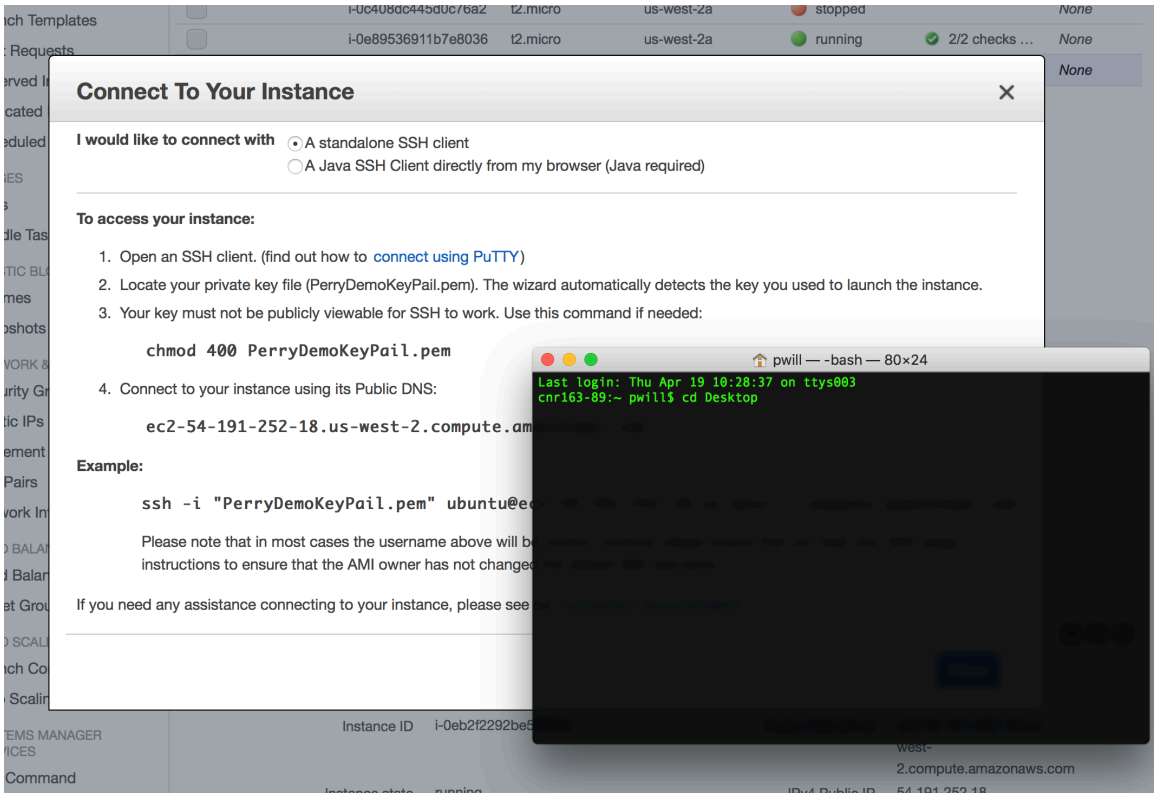


This brings up a terminal window:



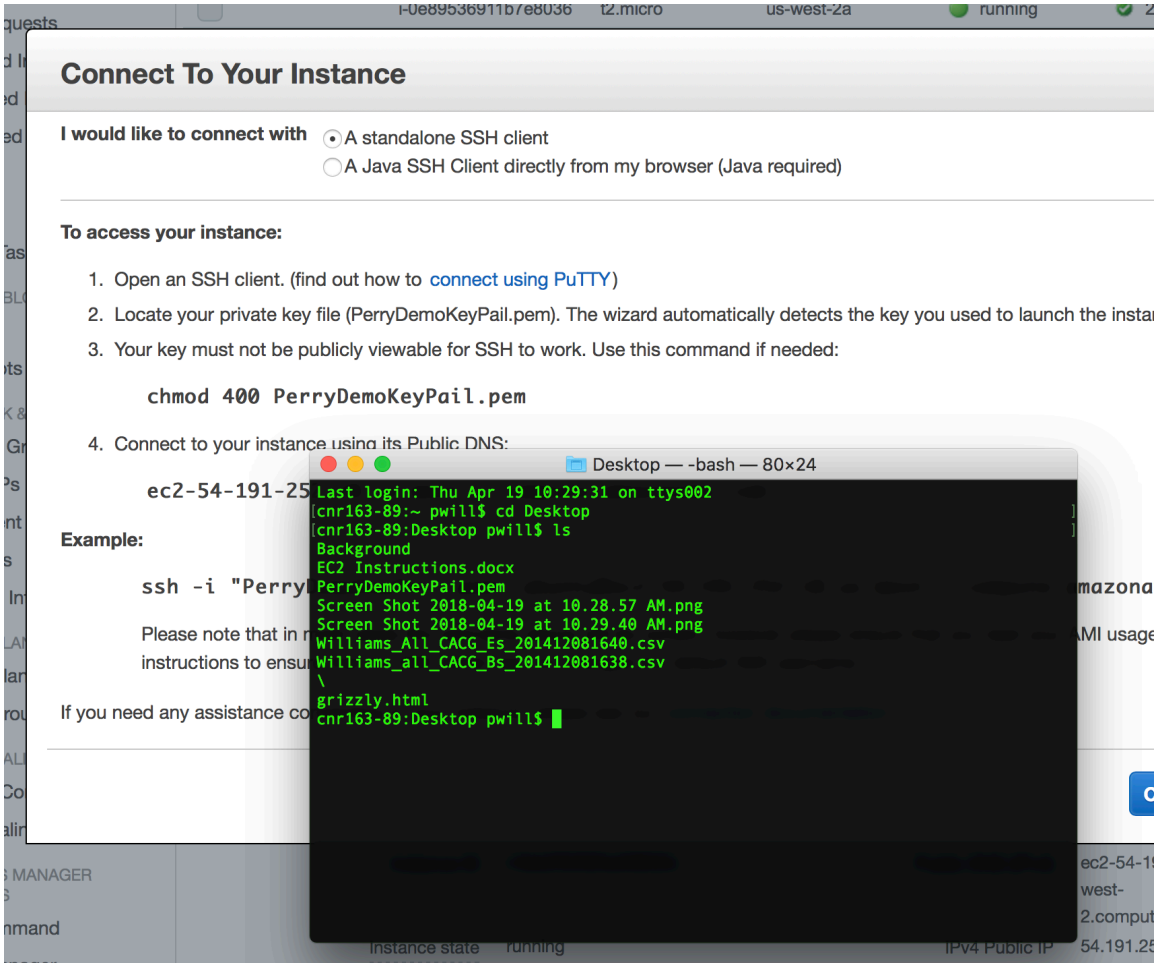
2. Locate your private key file.
I stored the private key file on the Desktop so type:
“cd Desktop” in the terminal:

```
pwill$ cd Desktop
```



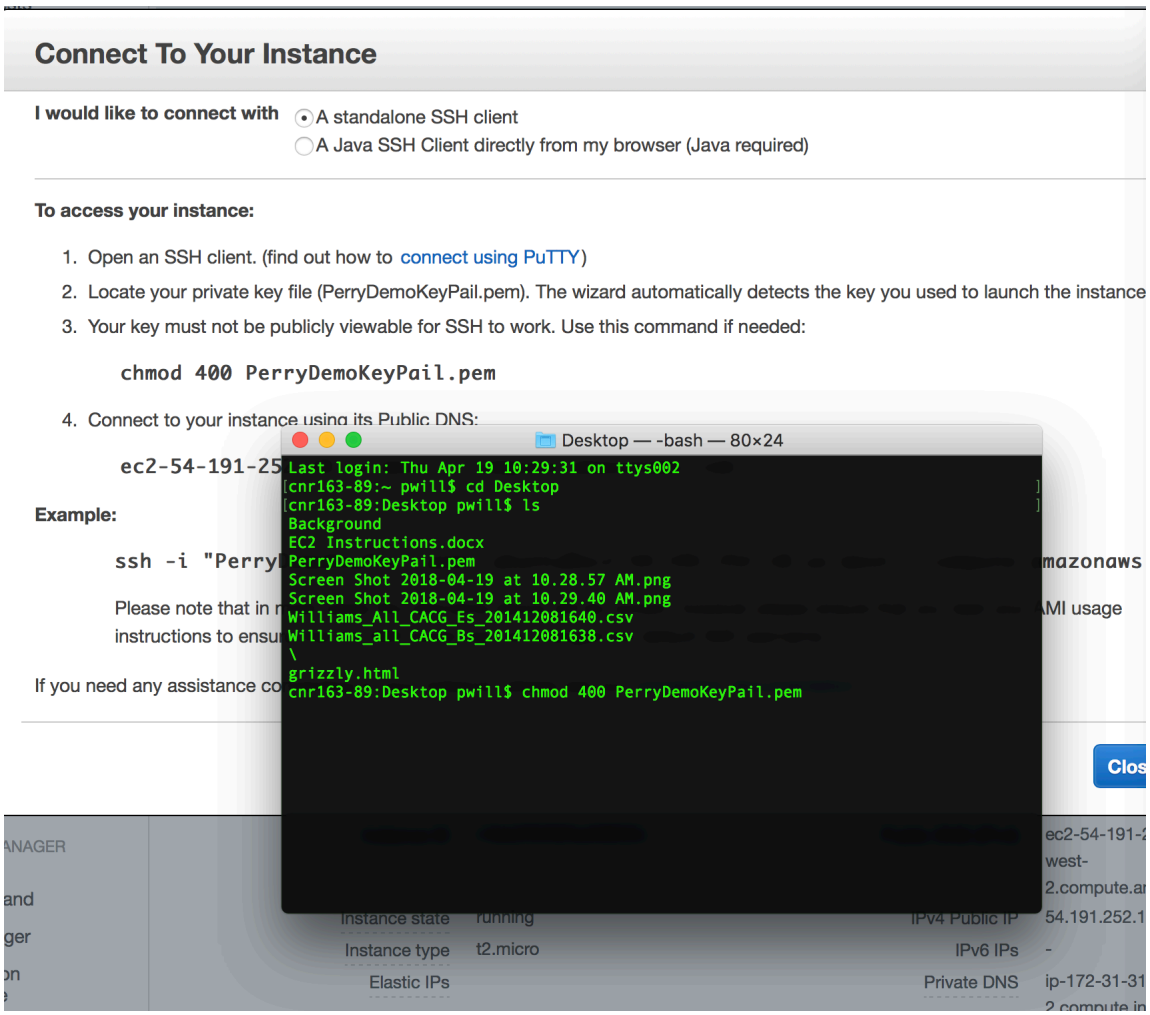
Look at files on the desktop and notice .pem file is there.

Desktop pwill$ ls



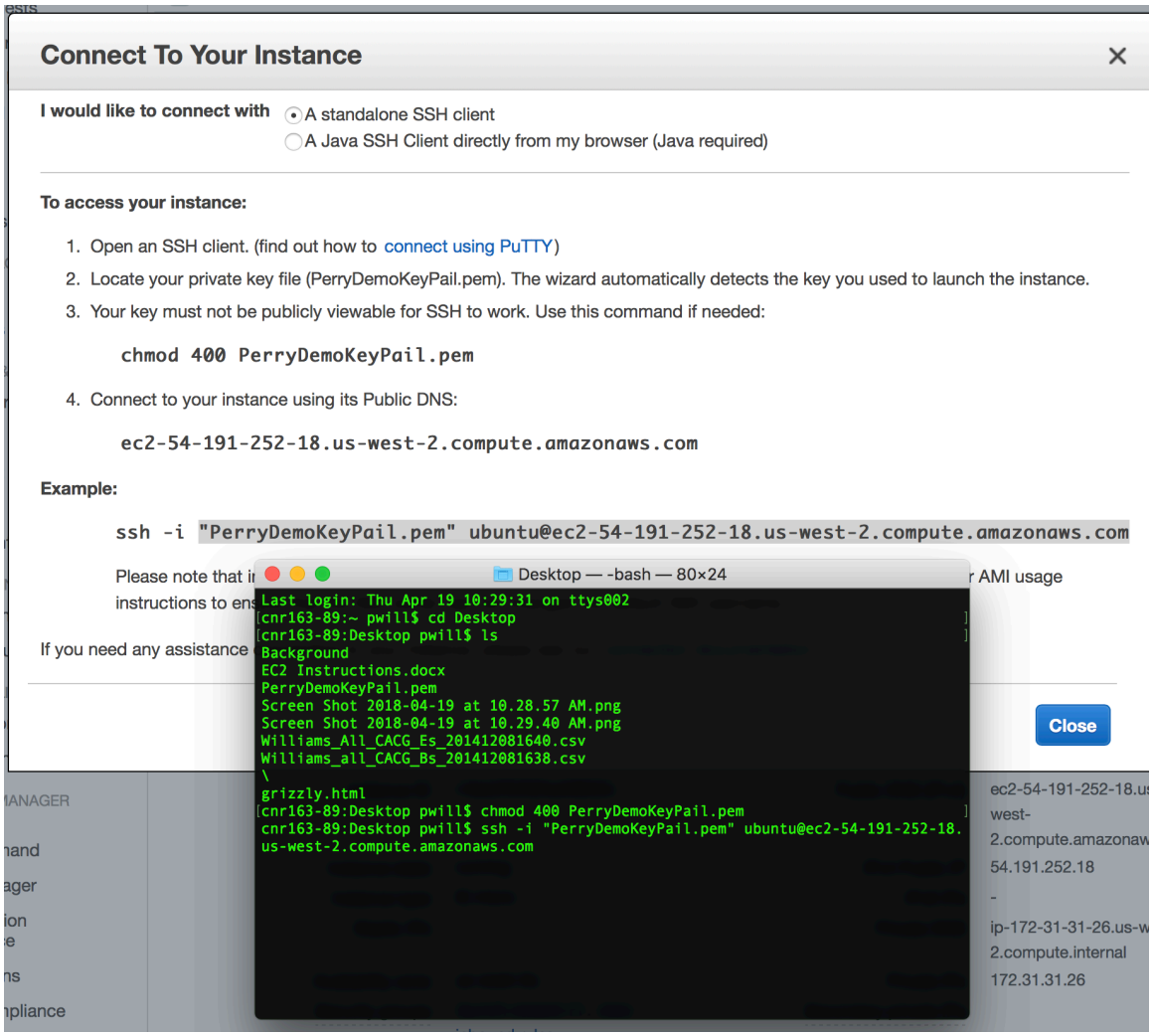
3. Run chmod 400 PerryKeyPailApril2018.pem in terminal (or whatever the keypair is).

```
Desktop pwill$ chmod 400 PerryKeyPailApril2018.pem
```

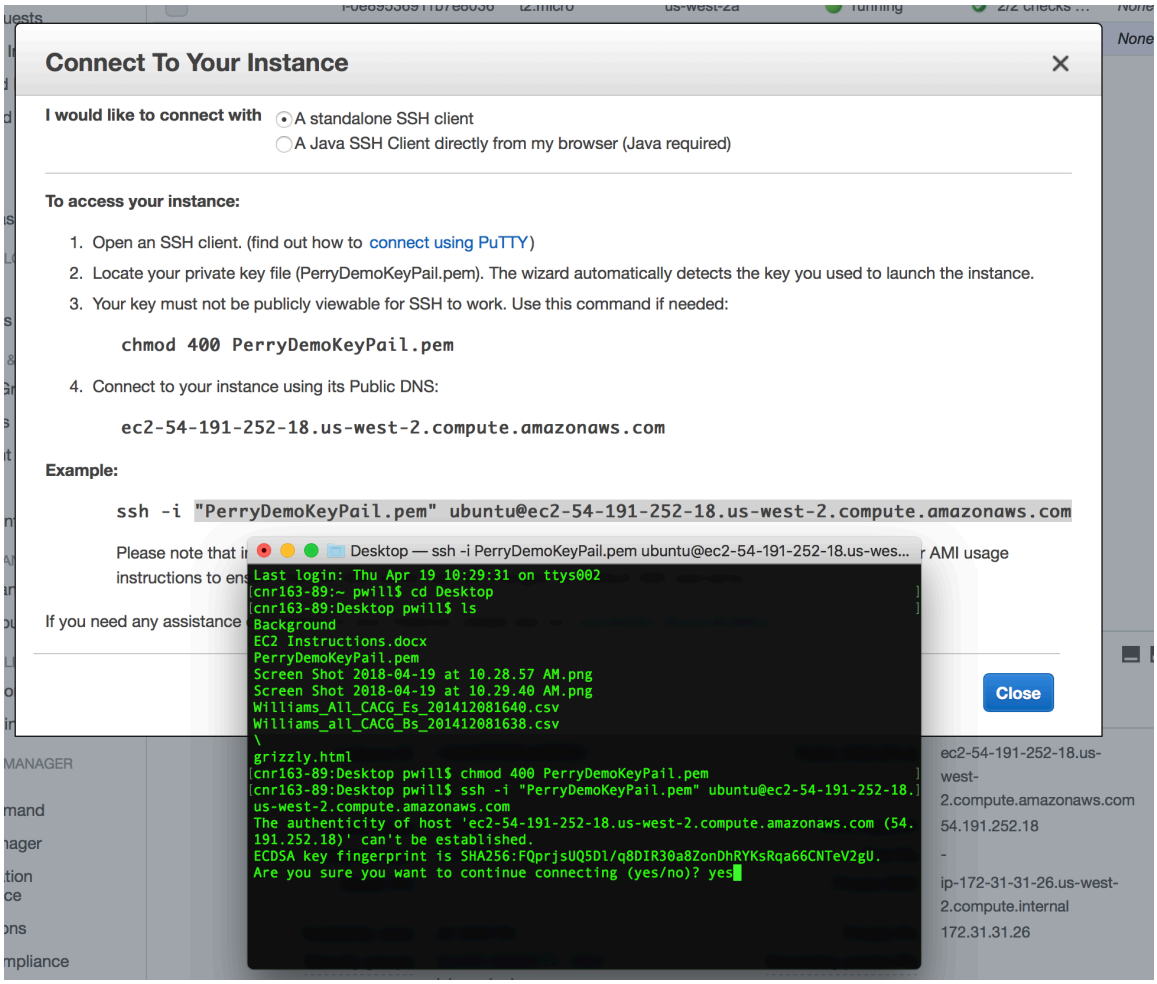


4. Connect to your instance using its Public DNS:

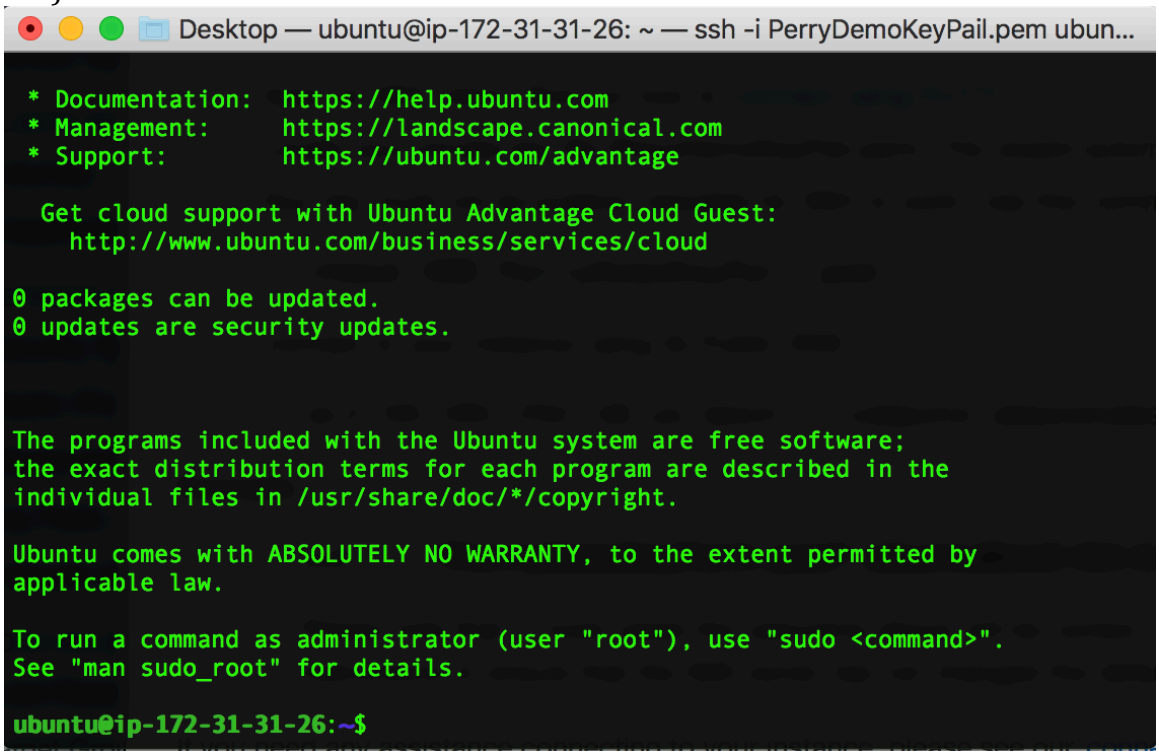
```
Desktop pwill$ ssh -i "PerryKeyPailApril2018.pem"
ubuntu@ec2-34-217-111-216.us-west-2.compute.amazonaws.com
```



Type in “yes” and push return
Desktop pwill $yes



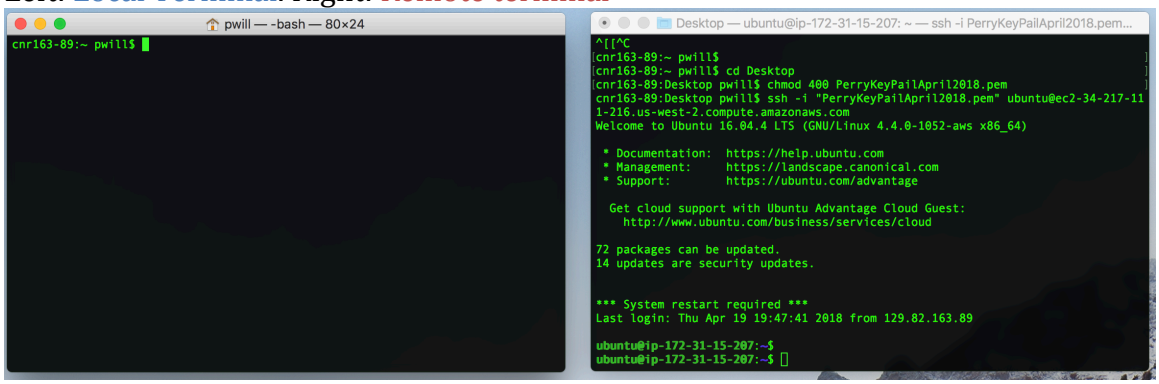
Our terminal is now connected to the instance (keep this terminal open for later use).



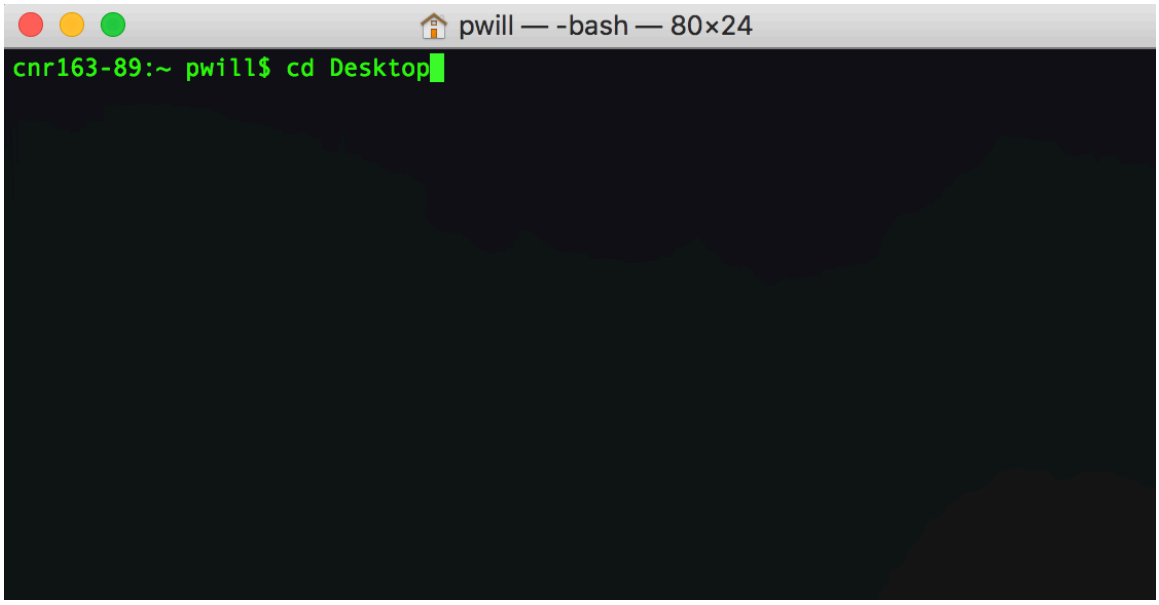
Transfer files to instance—I want to transfer my files to process on the remote instance. I'm going to use "scp" (secure-copy-paste)

First open new terminal (we'll put this side-by-side with the previous terminal connected to the remote instance):

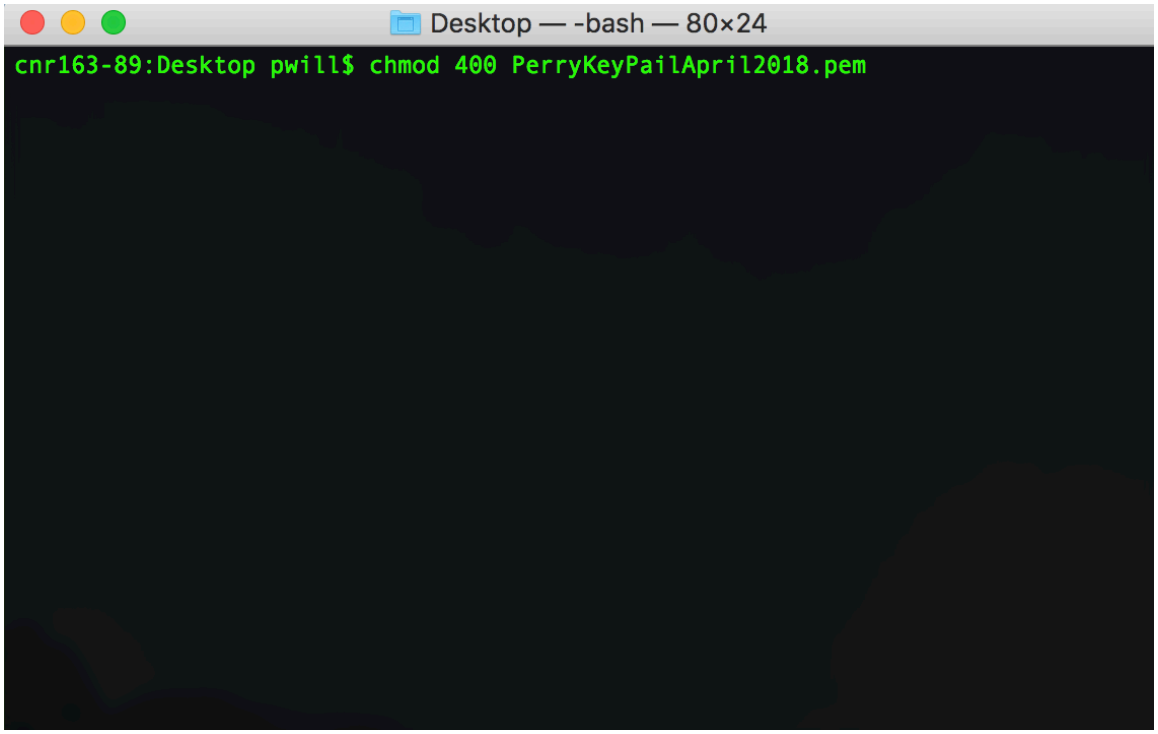
Left: Local Terminal. Right: Remote terminal



Next, on the Local Terminal, navigate to where key pair is stored:



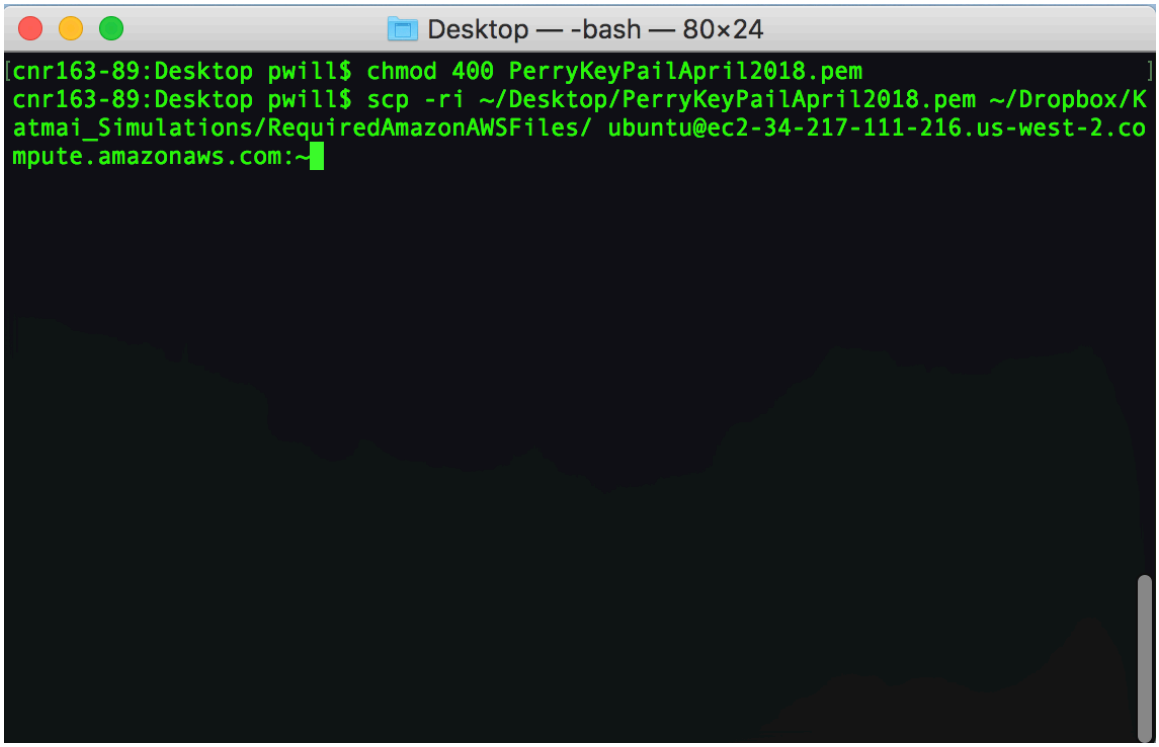
Make sure it is secure.



Next use ‘scp’ to move the files we will need from local directory to remote instance:

```
Desktop pwill$ scp -ri ~/Desktop/PerryKeyPailApril2018.pem
~/Dropbox/Katmai_Simulations/RequiredAmazonAWSFiles/
ubuntu@ec2-34-217-111-216.us-west-2.compute.amazonaws.com:~
```

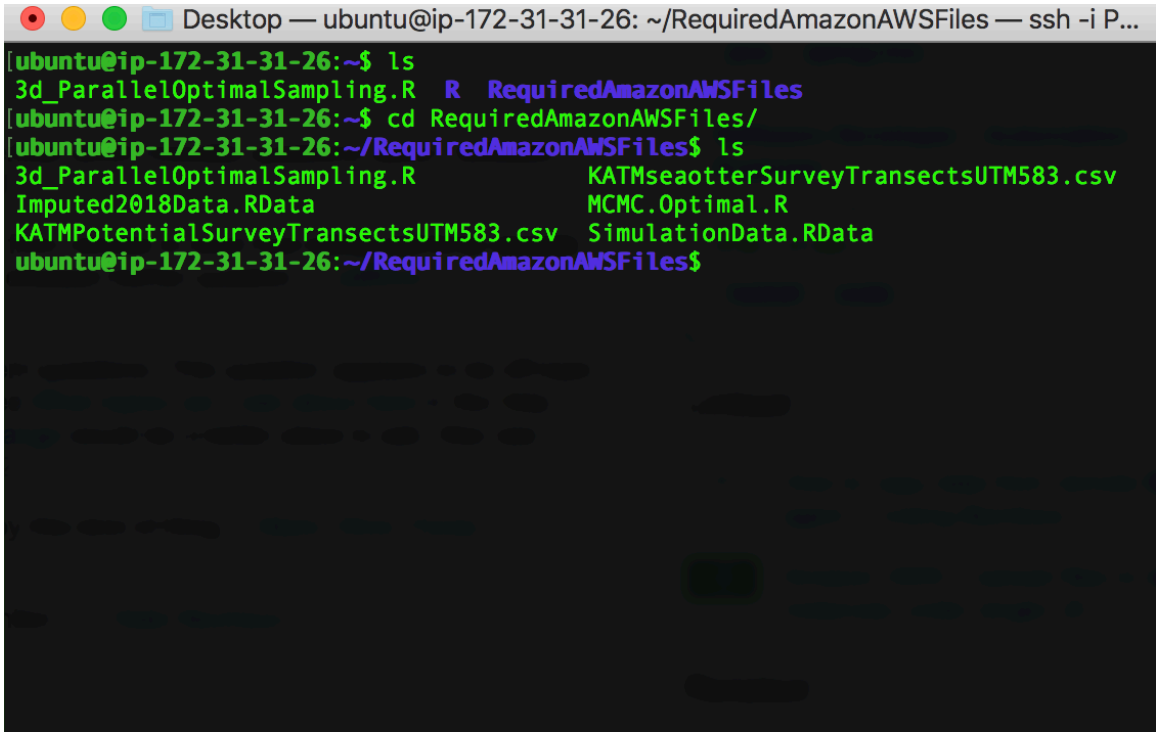
Local Terminal



Go back to the other terminal that we kept open to make sure the file transfer worked. Type “ls” and make sure the file is stored on the remote instance. The files in “RequiredAmazonAWSFiles” are now located in the remote instance.

```
ubuntu@ip-172-31-15-207:~$ ls
R  RequiredAmazonAWSFiles
ubuntu@ip-172-31-15-207:~$ cd RequiredAmazonAWSFiles/
ubuntu@ip-172-31-15-207:~/RequiredAmazonAWSFiles$ ls
3d_ParallelOptimalSampling.R
KATMseaotterSurveyTransectsUTM583.csv
Imputed2018Data.RData                    MCMC.Optimal.R
KATMPotentialSurveyTransectsUTM583.csv
SimulationData.RData
```

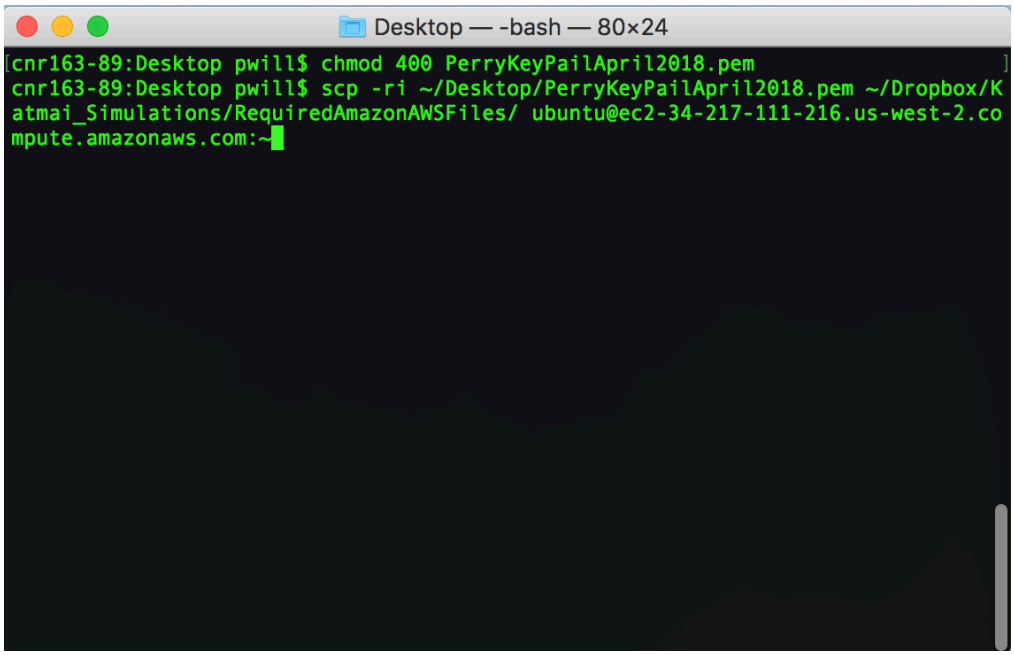
Remote terminal



Now we need to install R on the remote instance.

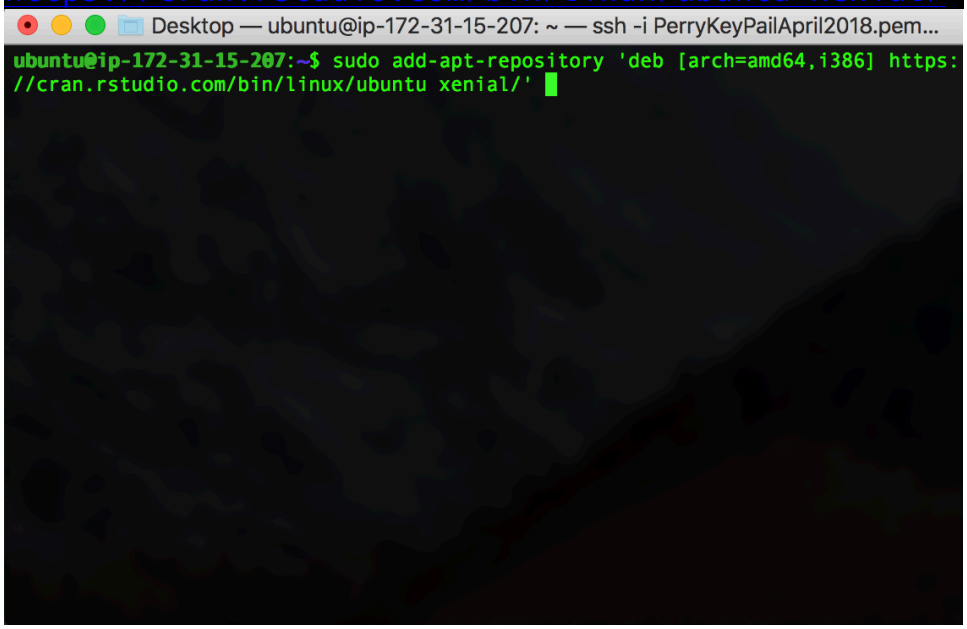
Type in the Remote terminal:

```
sudo apt-key adv --keyserver keyserver.ubuntu.com --recv-keys E298A3A825C0D65DFD57CBB651716619E084DAB9
```

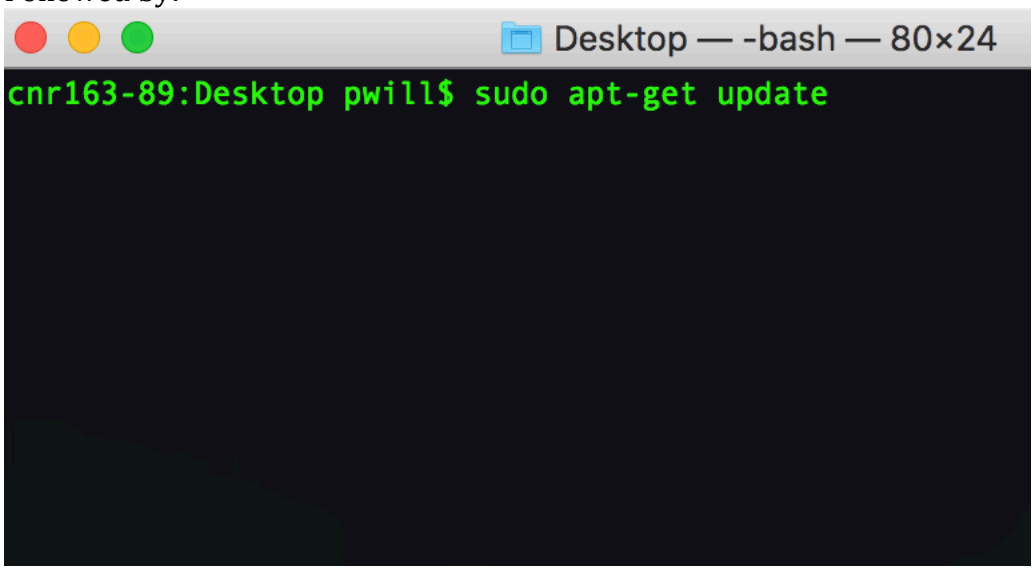


Followed by:

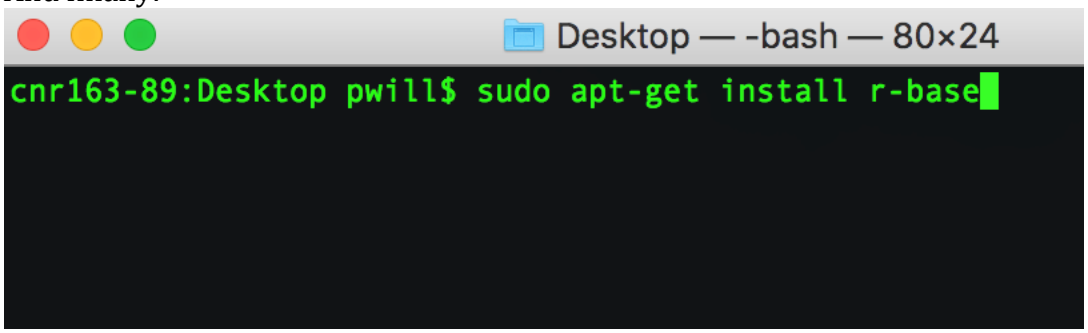
```
sudo add-apt-repository 'deb [arch=amd64,i386] https://cran.rstudio.com/bin/linux/ubuntu xenial/'
```



Followed by:

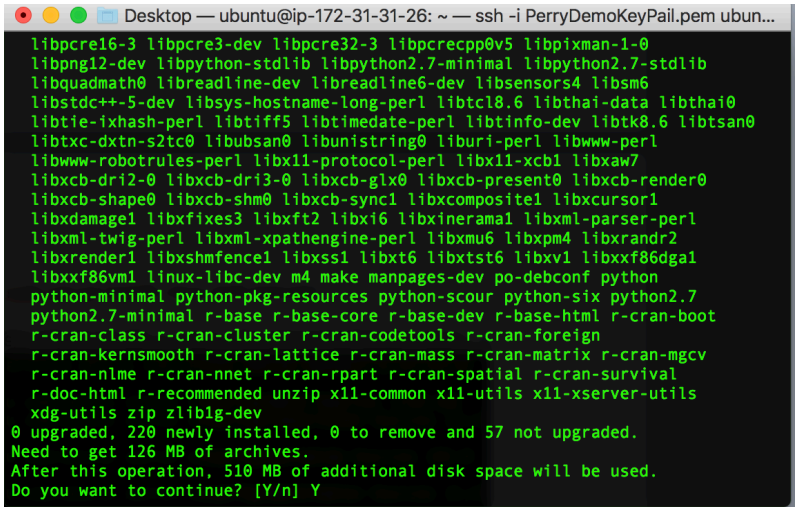


And finally:



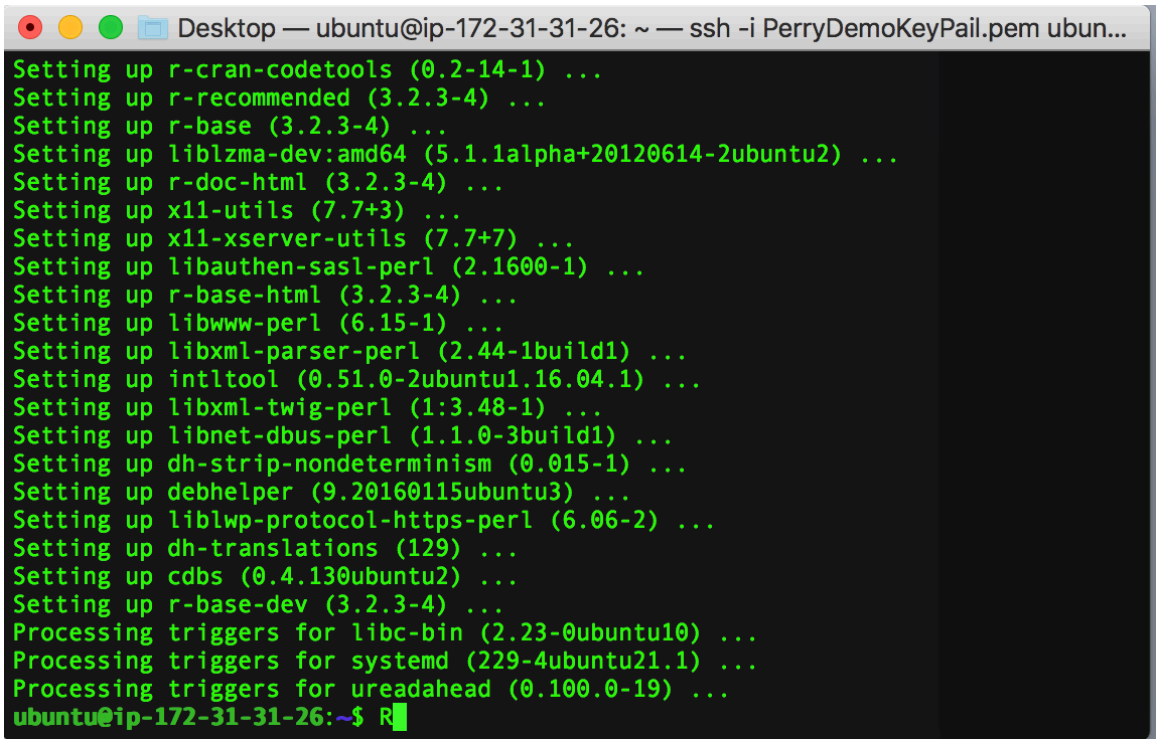
Yes, you want to continue:

Remote Terminal

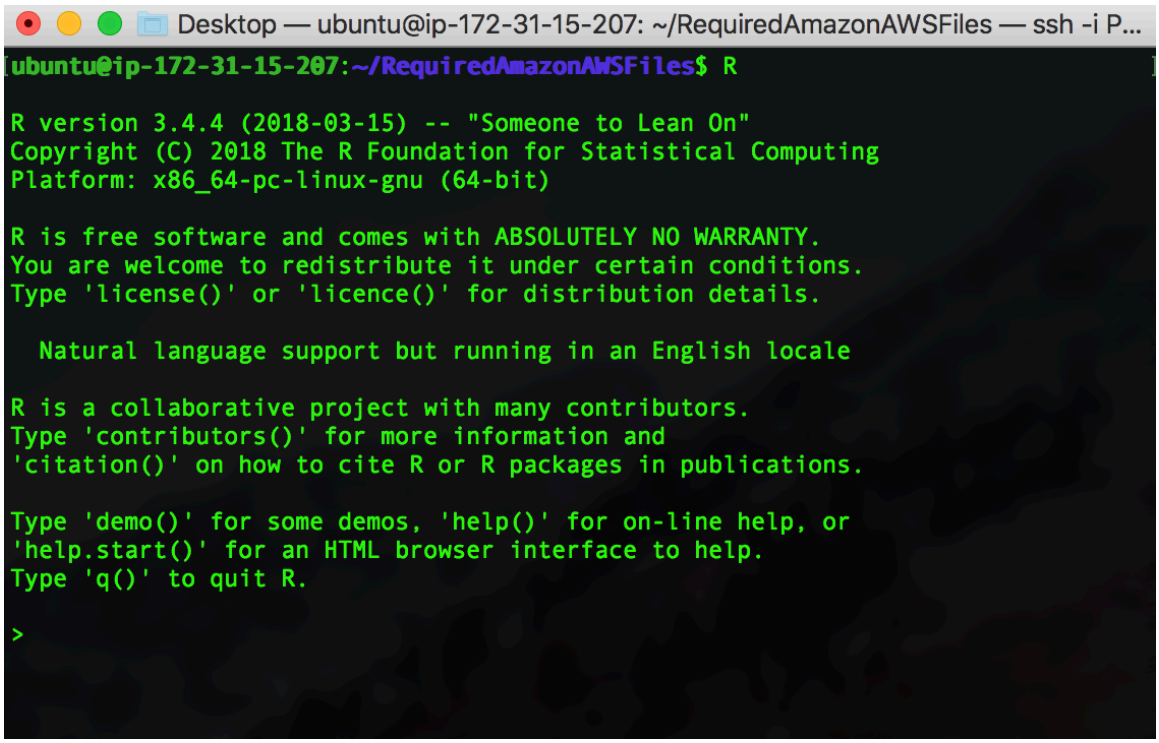


Check is R installed correctly by opening it:

Remote Terminal



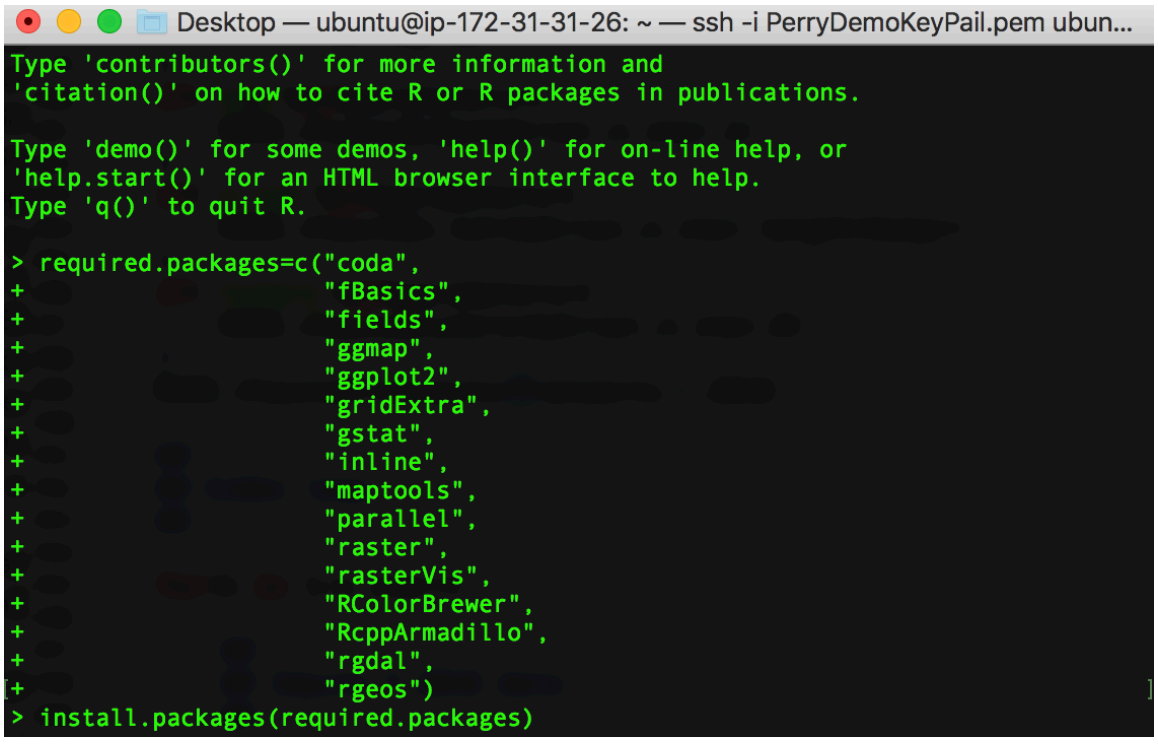
Remote Terminal



While we have R open, lets install the required packages for our specific analysis:

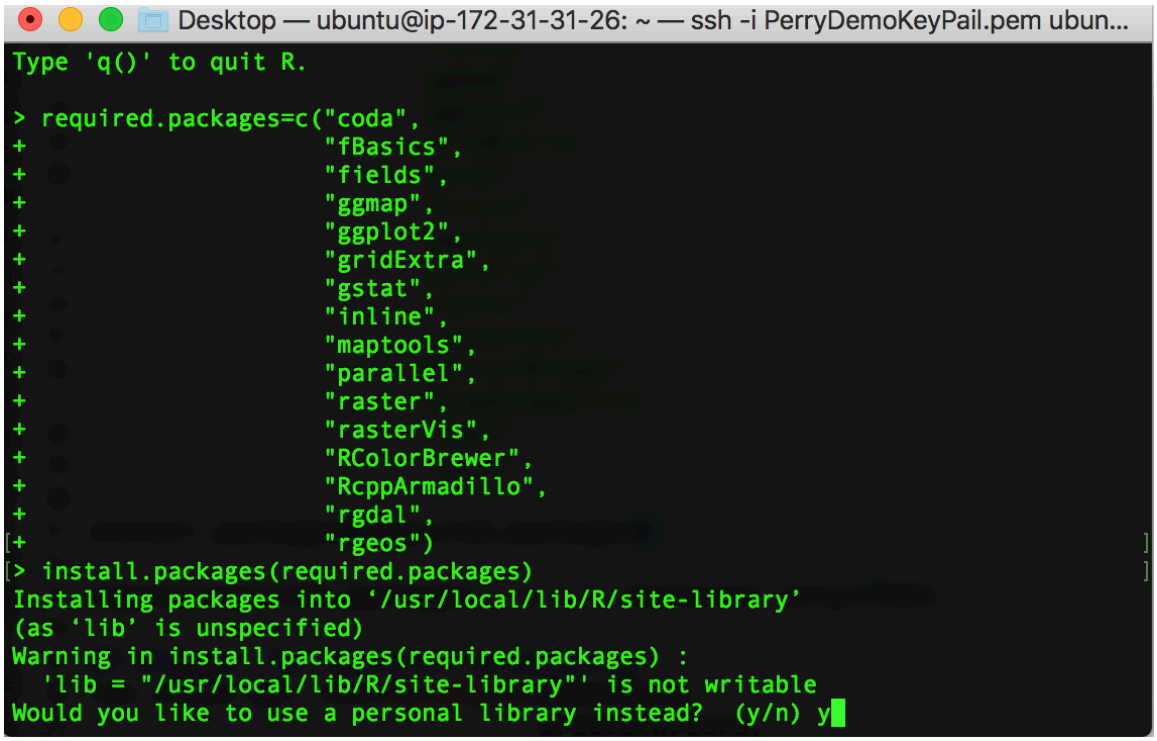
```
> required.packages=c("coda",
                      "fBasics",
                      "fields",
                      "ggmap",
                      "ggplot2",
                      "gridExtra",
                      "gstat",
                      "inline",
                      "maptools",
                      "parallel",
                      "raster",
                      "rasterVis",
                      "RColorBrewer",
                      "RcppArmadillo",
                      "rgdal",
                      "rgeos")

> install.packages(required.packages)
```

Remote Terminal



Hit “y” twice, and select a CRAN mirror. This will take a minute.

Remote Terminal



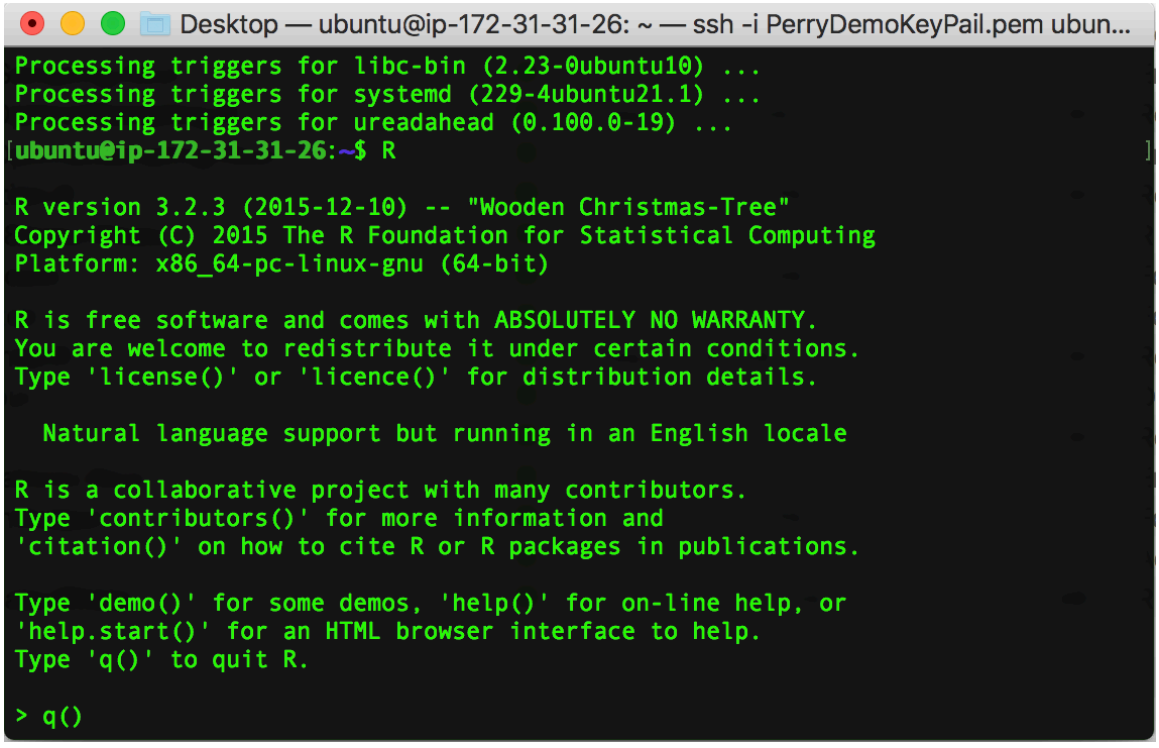
After the packages are installed try:

> detectCores()

to see how many cores we now have available for parallel processing.

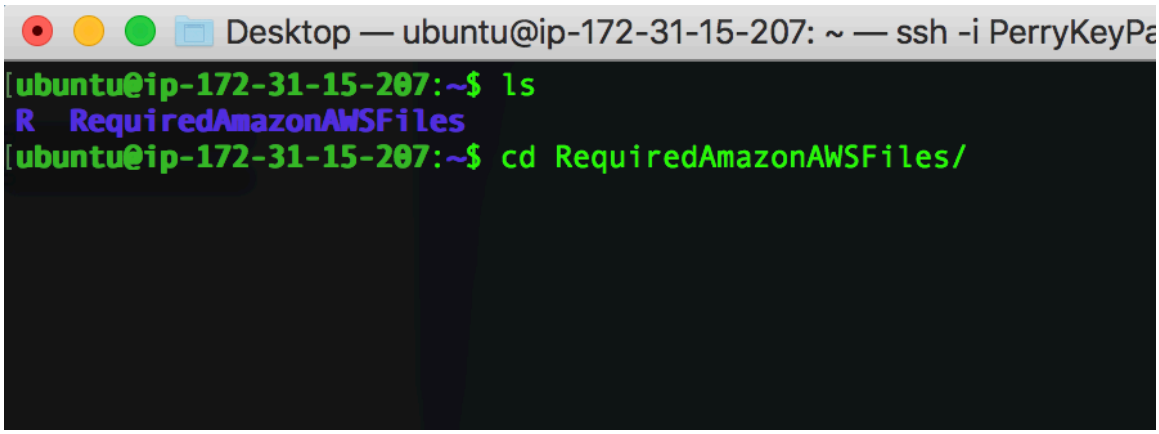
Remote Terminal

Quit R (we're going to use BATCH scripts to run R).



The final step is to run the script for parallel processing. Our script is entitled: "3d_ParrallelOptimalSampling.R"

On the Remote terminal, after exiting R, navigate to where the file is stored:



Run R CMD BATCH script to run the entire script, which saves the output.

Last, we need to transfer the results to our local drive (The output will be huge):