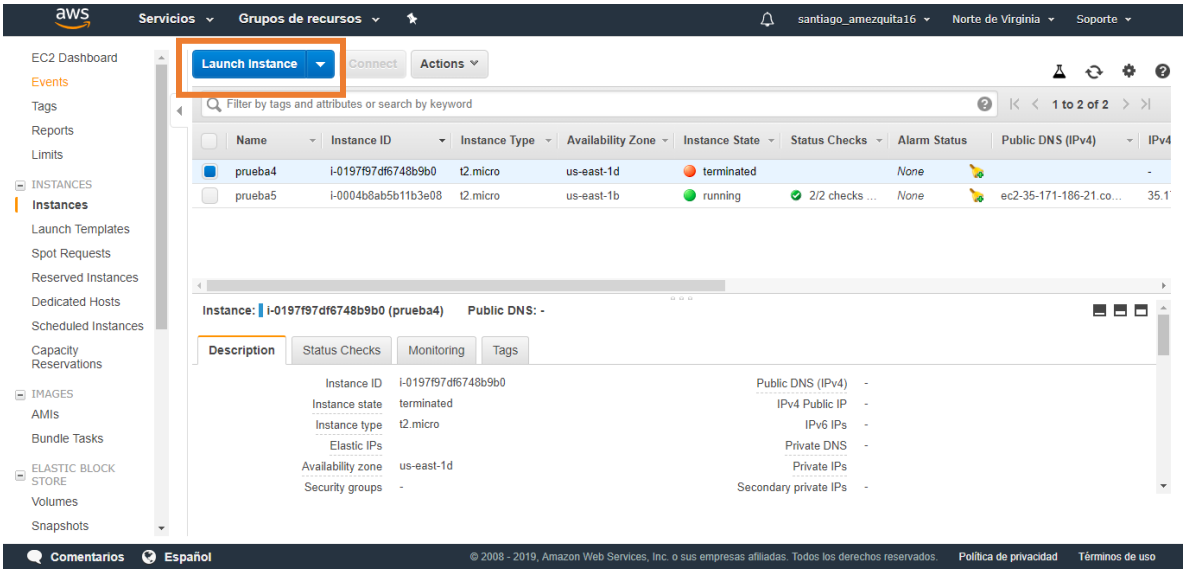
Uploading a shiny-server in AWS

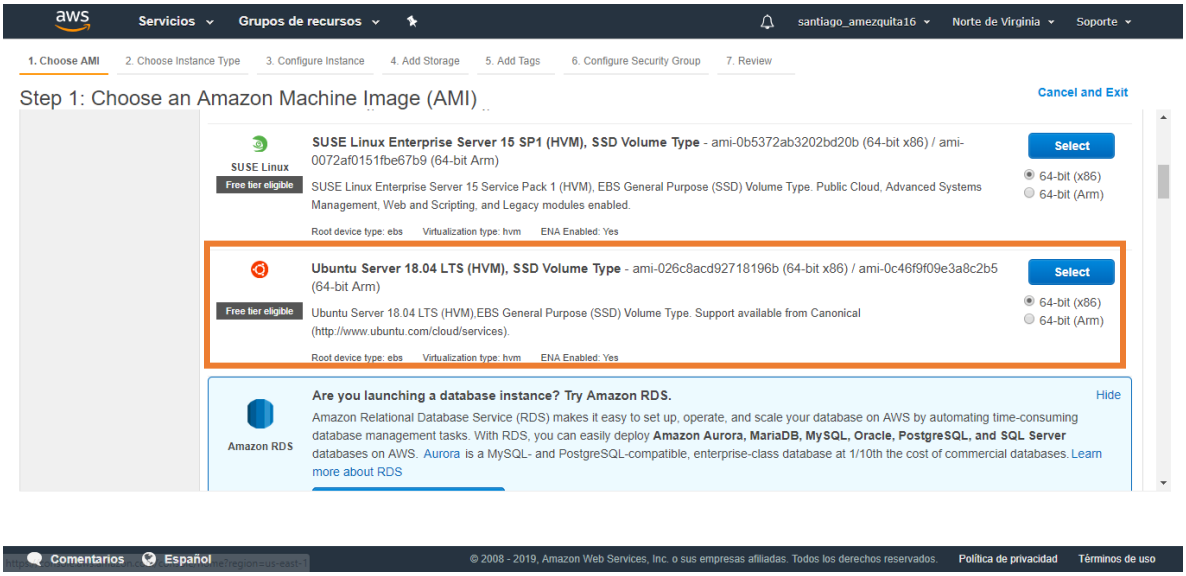
The following documentation is a step-by-step guide to upload a shiny server on AWS.

Creating the AWS server

1. Log into Amazon Web Services (AWS from now on).
2. In the "Services" section, look for EC2 to create a server
3. Click on launch instance:

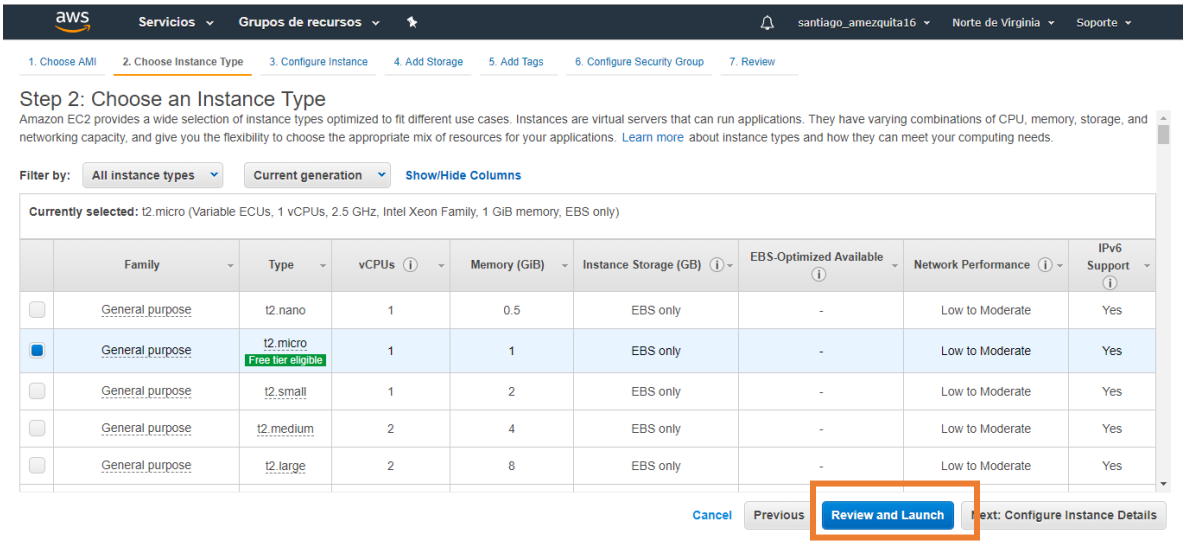


4. Look for a free tier eligible server, we will use in this case an Ubuntu Server 18.04 LTS of 64-bits.

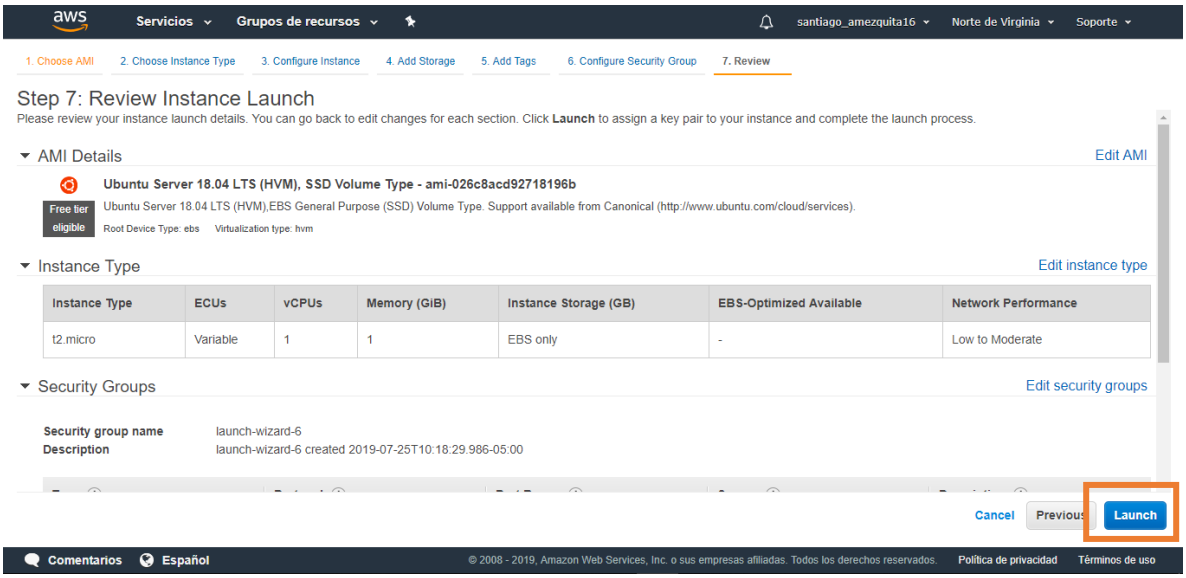


EC2 allows you to create elastic servers, which can be used for simple tasks of the shiny apps or "enlarge" themselves to do more complex tasks.

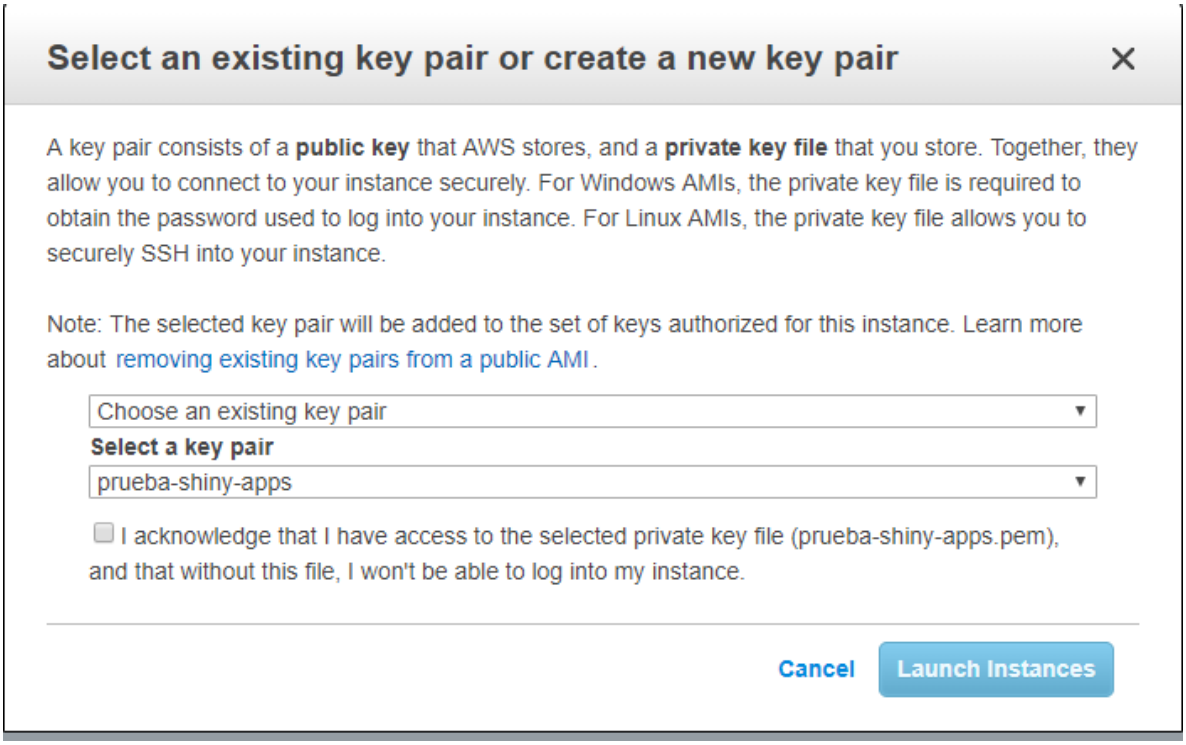
5. Once you choose the Amazon Machine Image (AMI from now on), click on “Review and Launch”,



6. Click on “Launch”.



7. Once you click on launch, you should get something like this:

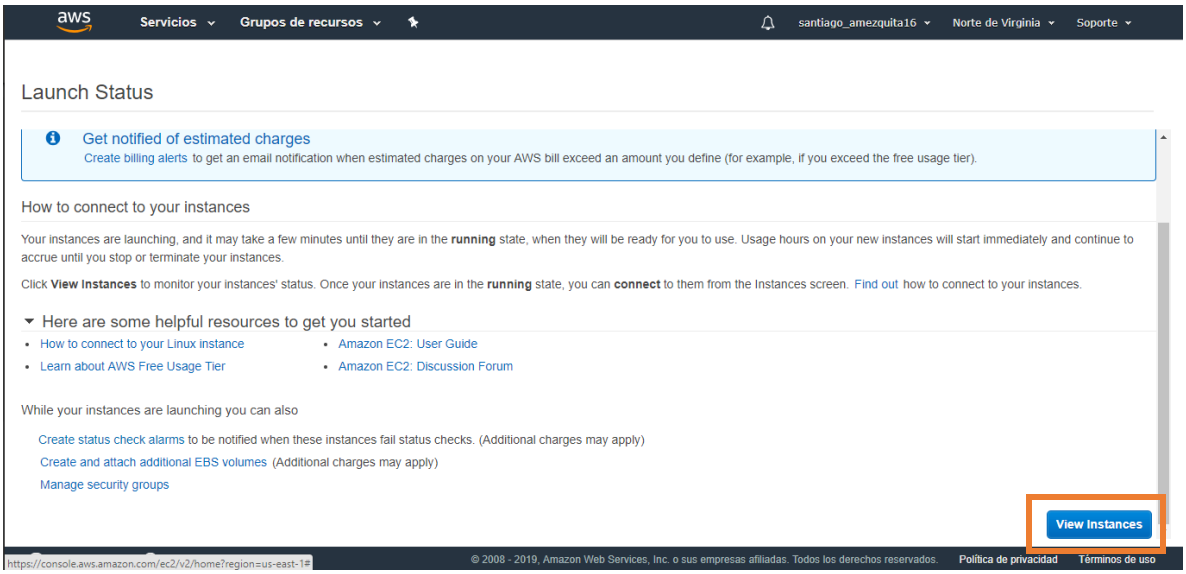


This is to create a key for your AMI and access to it remotely. It will save a .pem file. **DO NOT LOOSE THIS FILE,** without it you cannot access your AMI in the future. Since you probably don't have a key pair, you should create one:

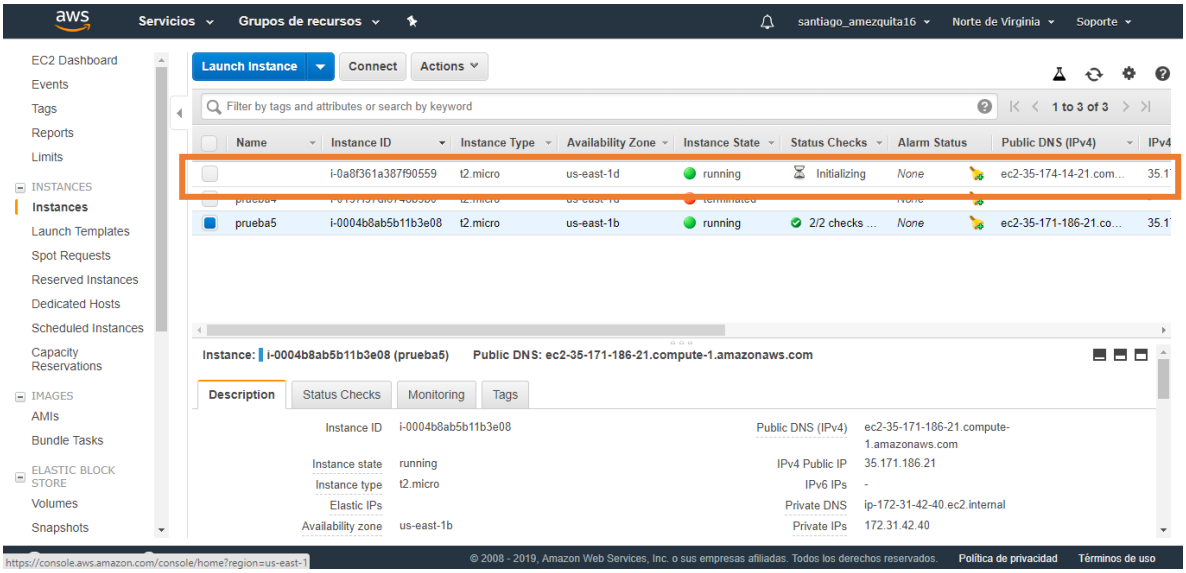
7.1. In the first list select "Create a new key pair"
7.2. Create a simple name separating words with "-".
7.3. Download the key pair and **SAVE IT IN A SAFE AND EASY TO ACCESS PLACE.**

In this case, we will use an existing key pair called "prueba-shiny-apps". If you're going to use an existing key pair, check the box on the bottom of this screen and click on "Launch Instances".

8. In the bottom of this new page, click on "View Instances" to see all of the instances that you may have.



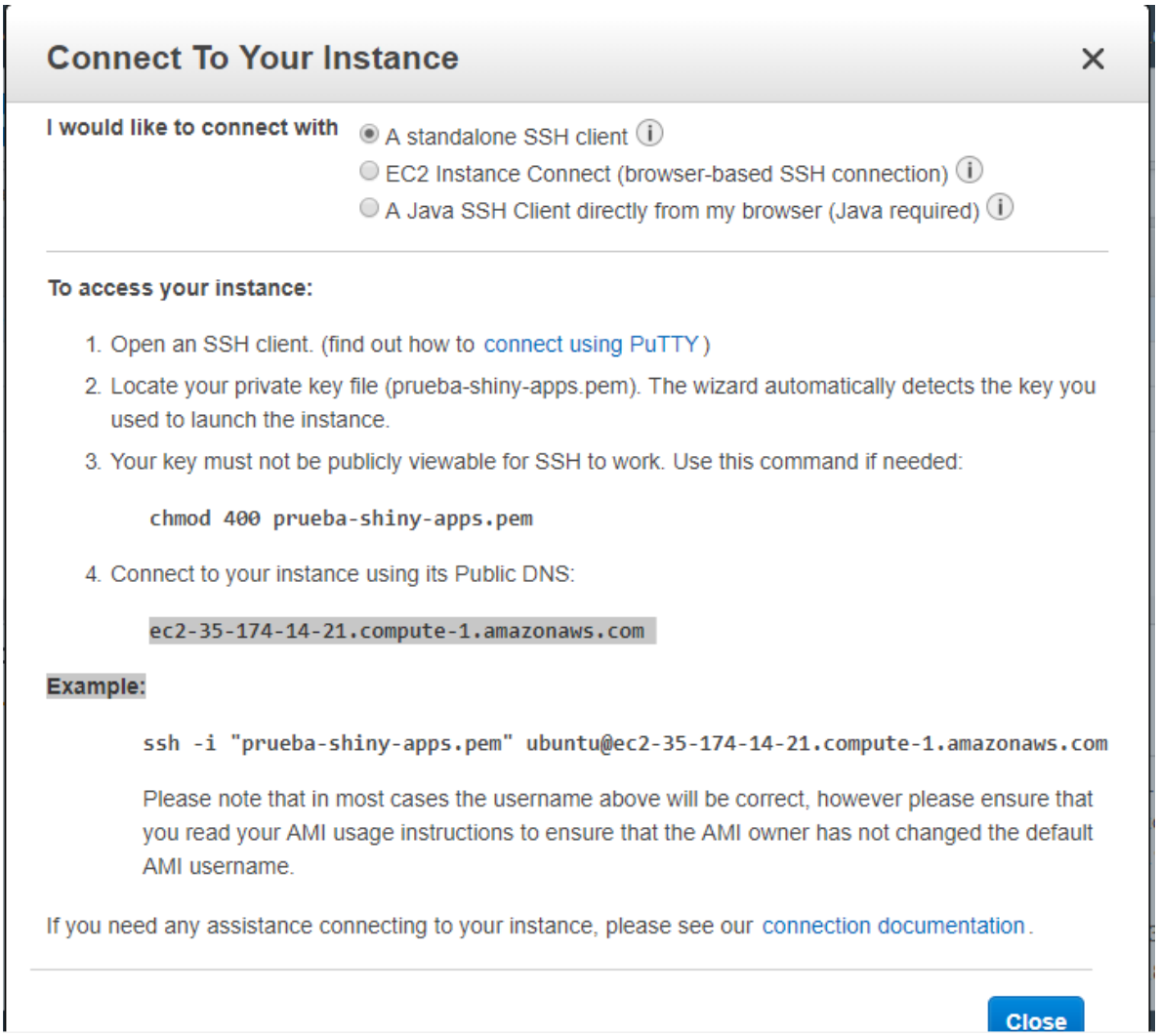
Now, we can see our new instance with no name. If you want to, you can change the name of the instance.



We will change the name in this tutorial to “instance1”.

If we want to access from a windows machine, we will have to download PuTTy to access this machine. AWS contains the whole documentation with this process.

To connect with PuTTy, select the instance in AWS and click on “Connect”, you’ll get something like this:



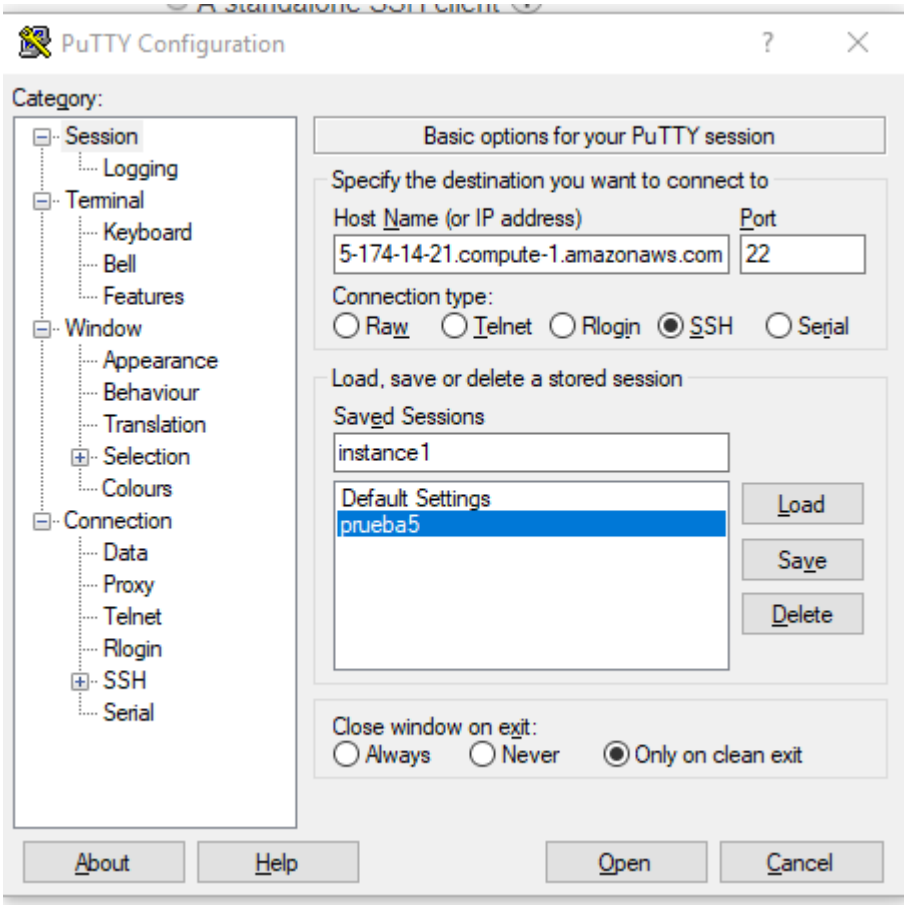
The selected text is what we will need to access to this machine in PuTTy.

9. Open PuTTy and create the server:

9.1. In the "Session" page, copy and paste your Instance's public DNS, in my case, it is "ec2-35-174-14-21.compute-1.amazonaws.com".

9.2. In order to have an organized environment in PuTTy, save the instance name with the same name you have on AWS, in the case of this tutorial, "instance1".

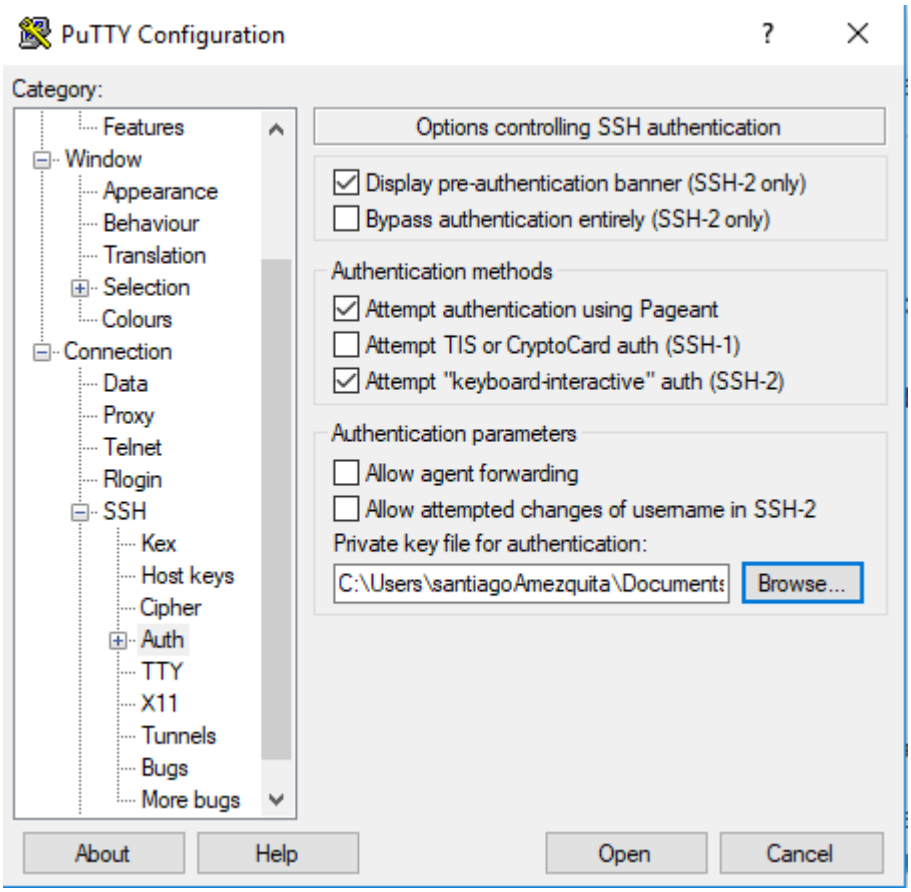
9.3. Click on Save. You'll have a PuTTy screen like this:



Check that all the sections in the "Session" screen are almost equal to the ones showed here (Select SSH, select "Only on clean exit").

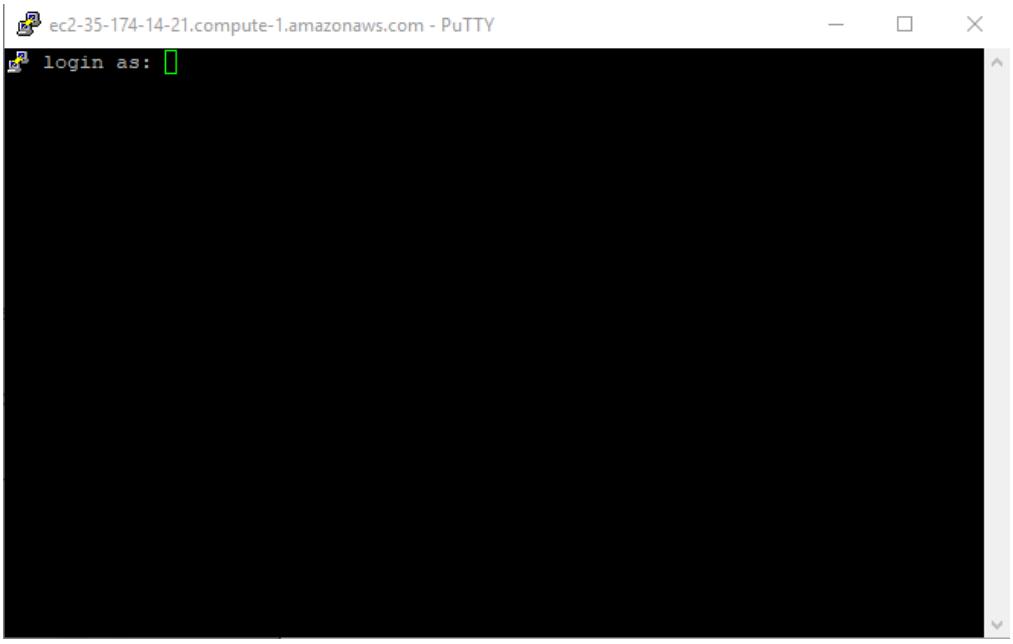
9.4. Now we have to connect to the machine with the key pair we downloaded previously, with a .ppk extension. This will be done in the section Connection -> SSH -> Auth

9.5. Browse for your key pair and upload it, you'll get something like this:

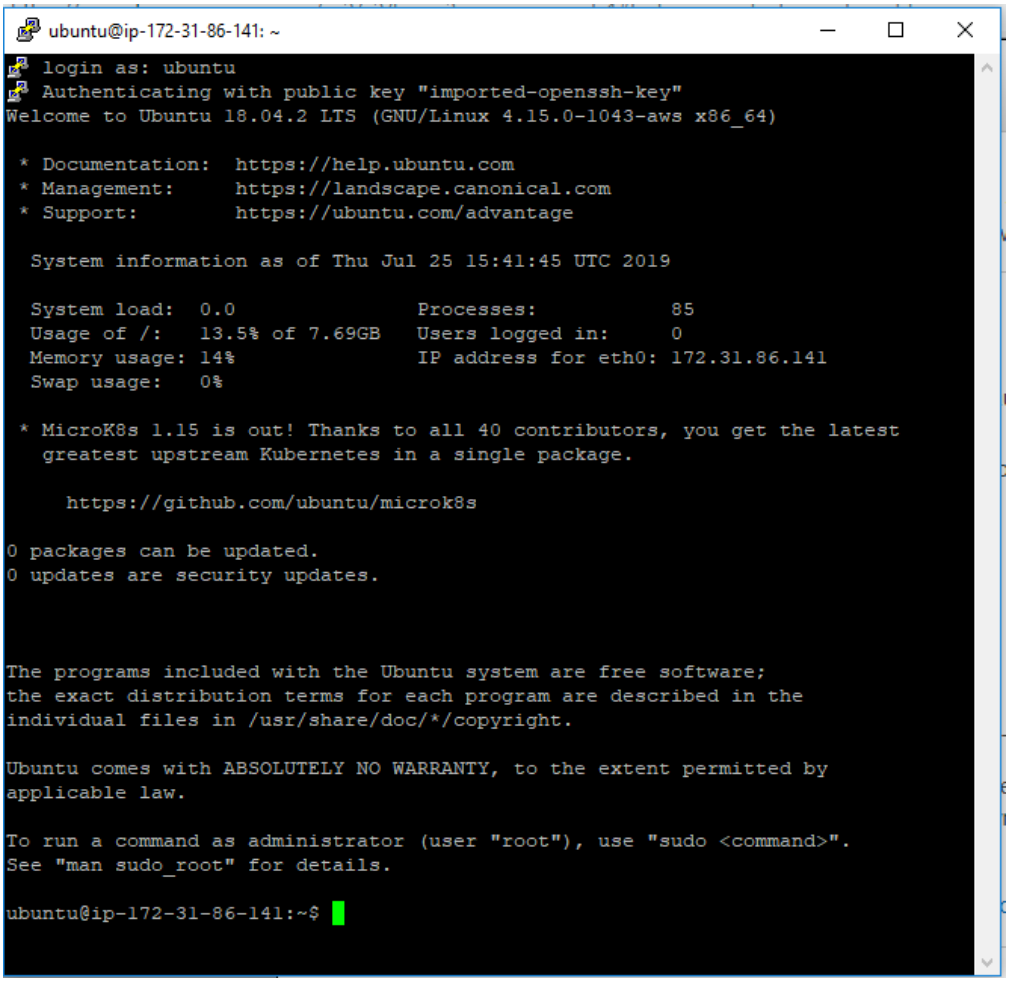


Again, check that everything has the same checkboxes checked.

9.6. Go the Session screen again and click on save again.
9.7. Click on “Open”.
9.8. Click on “Yes”.
10. When you successfully accessed to your AMI, you’ll get a screen like this one:



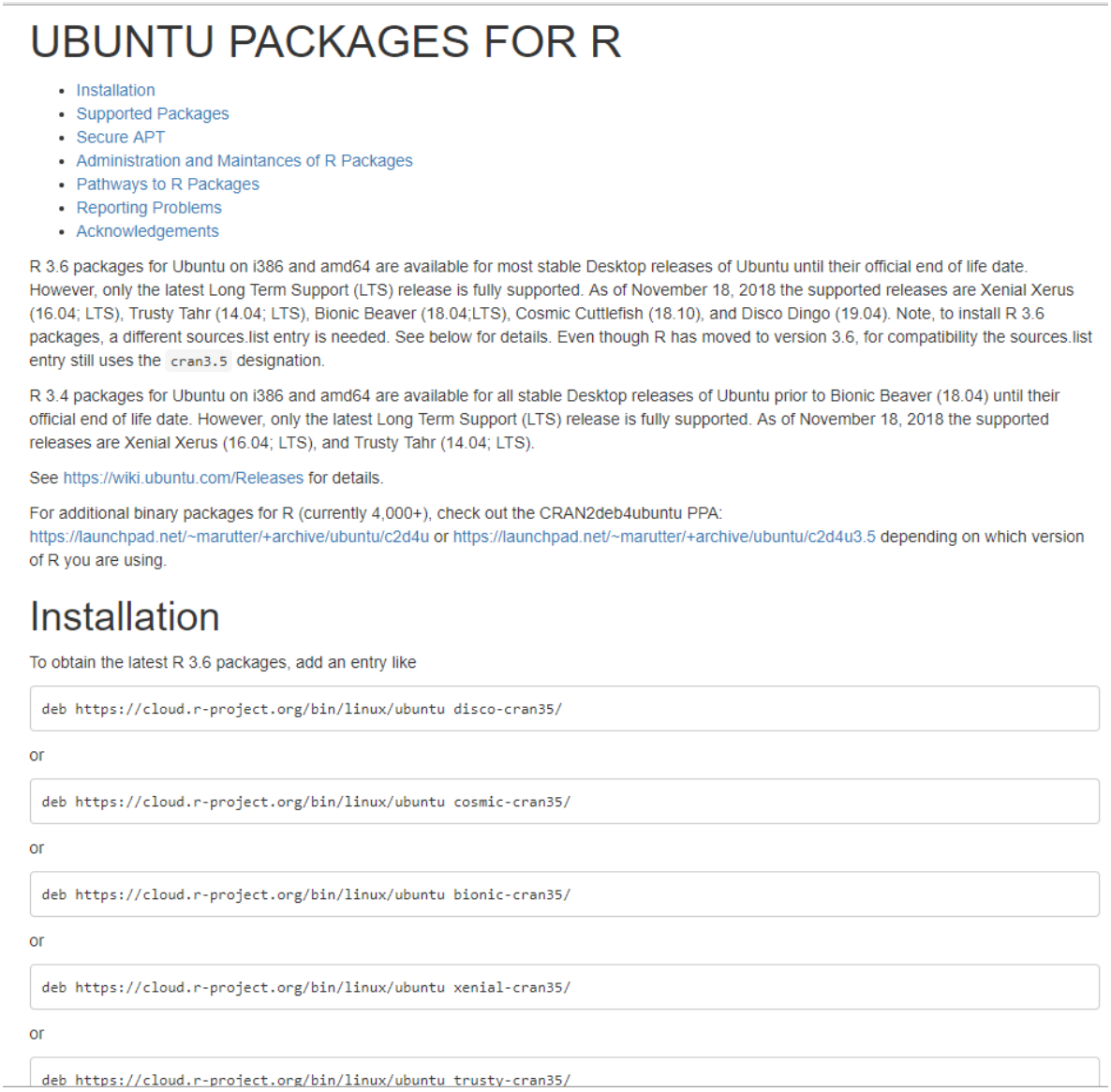
We will login as “ubuntu”, so, type ubuntu and press enter, you’ll get something like this:



Install R and Shiny in your AMI

Now we must install R and shiny.

11. The whole documentation to install R is on the link: https://cran.r-project.org/
11.1. Click on "Download R for Linux".
11.2. Click on "ubuntu".
11.3. You'll get a page like this:



UBUNTU PACKAGES FOR R

- Installation
- Supported Packages
- Secure APT
- Administration and Maintances of R Packages
- Pathways to R Packages
- Reporting Problems
- Acknowledgements

R 3.6 packages for Ubuntu on i386 and amd64 are available for most stable Desktop releases of Ubuntu until their official end of life date. However, only the latest Long Term Support (LTS) release is fully supported. As of November 18, 2018 the supported releases are Xenial Xerus (16.04; LTS), Trusty Tahr (14.04; LTS), Bionic Beaver (18.04;LTS), Cosmic Cuttlefish (18.10), and Disco Dingo (19.04). Note, to install R 3.6 packages, a different sources.list entry is needed. See below for details. Even though R has moved to version 3.6, for compatibility the sources.list entry still uses the `cran3.5` designation.

R 3.4 packages for Ubuntu on i386 and amd64 are available for all stable Desktop releases of Ubuntu prior to Bionic Beaver (18.04) until their official end of life date. However, only the latest Long Term Support (LTS) release is fully supported. As of November 18, 2018 the supported releases are Xenial Xerus (16.04; LTS), and Trusty Tahr (14.04; LTS).

See https://wiki.ubuntu.com/Releases for details.

For additional binary packages for R (currently 4,000+), check out the CRAN2deb4ubuntu PPA: https://launchpad.net/~marutter/+archive/ubuntu/c2d4u or https://launchpad.net/~marutter/+archive/ubuntu/c2d4u3.5 depending on which version of R you are using.

Installation

To obtain the latest R 3.6 packages, add an entry like

```
deb https://cloud.r-project.org/bin/linux/ubuntu disco-cran35/
```

or

```
deb https://cloud.r-project.org/bin/linux/ubuntu cosmic-cran35/
```

or

```
deb https://cloud.r-project.org/bin/linux/ubuntu bionic-cran35/
```

or

```
deb https://cloud.r-project.org/bin/linux/ubuntu xenial-cran35/
```

or

```
deb https://cloud.r-project.org/bin/linux/ubuntu trusty-cran35/
```

The whole documentation to install the latest version of R is on that page, however, we will do a step-by-step guide in this tutorial to install R successfully on your AMI.

12. Open the file sources.list located on the route /etc/apt/. To access this file, write the following lines and press enter in each one of them:

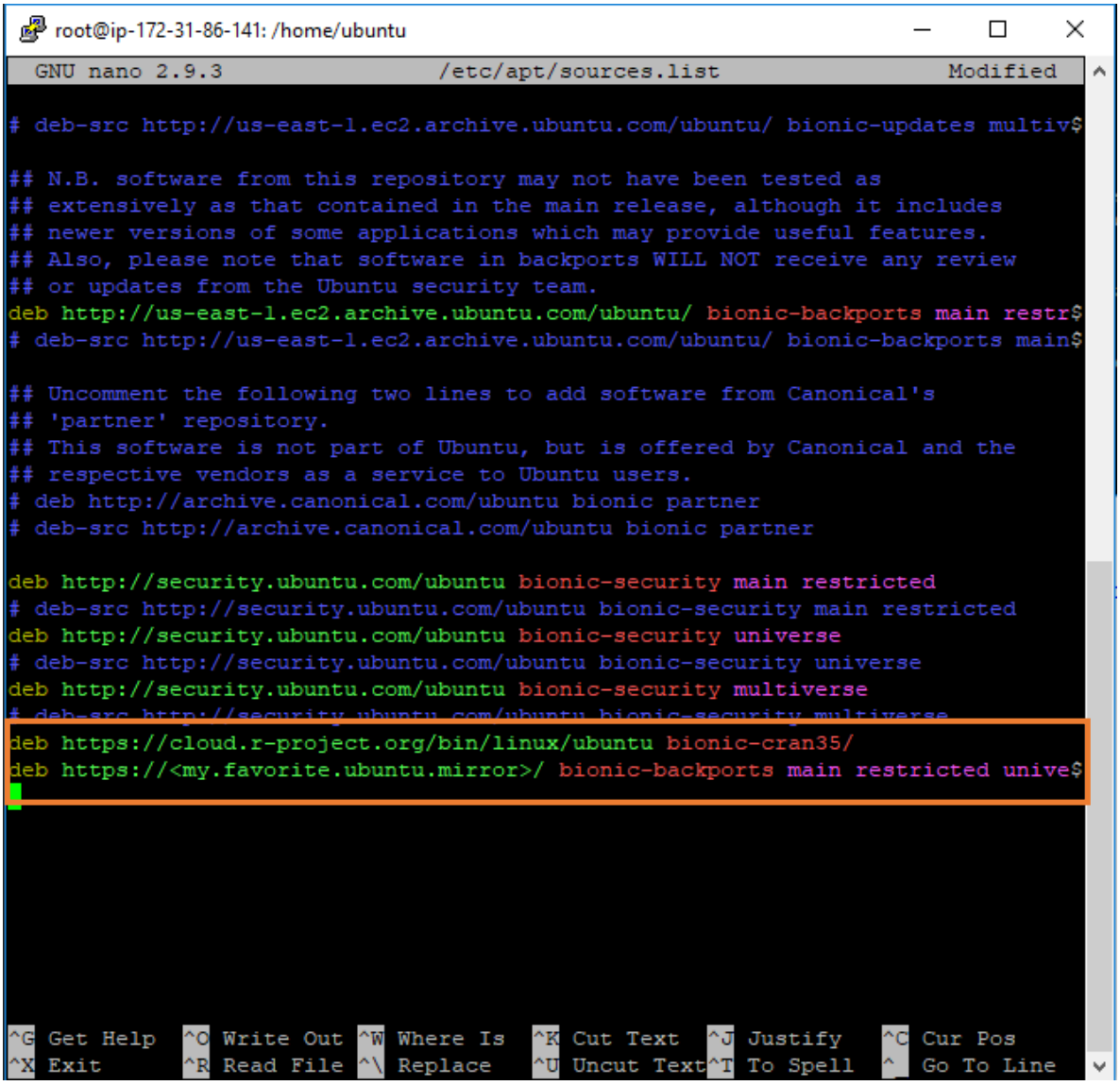
```
sudo su
```

```
nano /etc/apt/sources.list
```

Now that you're on the file, write the following lines in the bottom of the file:

```
deb https://cloud.r-project.org/bin/linux/ubuntu bionic-cran35/
deb https://<my.favorite.ubuntu.mirror>/ bionic-backports main restricted universe
```

And end up with a file like this one:



Press Ctrl+O

Press Enter

Press Ctrl+X

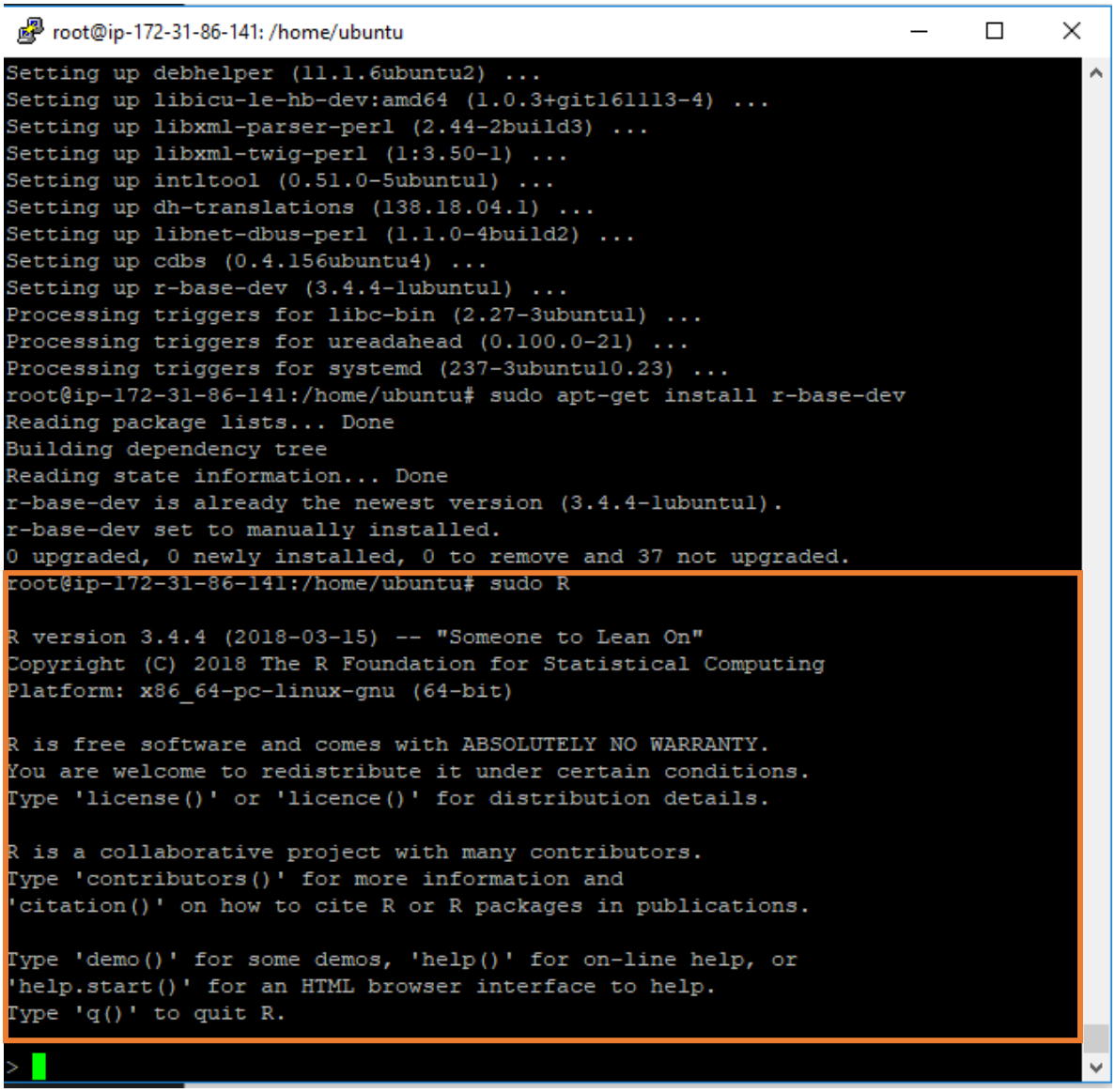
Now you must put these three commands in your AMI's console:

```
sudo apt-get update
sudo apt-get install r-base
sudo apt-get install r-base-dev
```

To access R from your AMI you can write:

```
sudo R
```

And you'll get something like this:



At this point we have to install shiny directly from R with the following line:

```
install.packages("shiny")
```

If for some reason this doesn't works or takes too long, abort the task with Ctrl+C, quit R and use the command:

```
sudo apt-get install r-cran-shiny
```

This last command will be very important to install the R packages in case it doesn't works directly in R.

To exit from R console, you can type q() or press Ctrl+Z.

Configure your Shiny-server

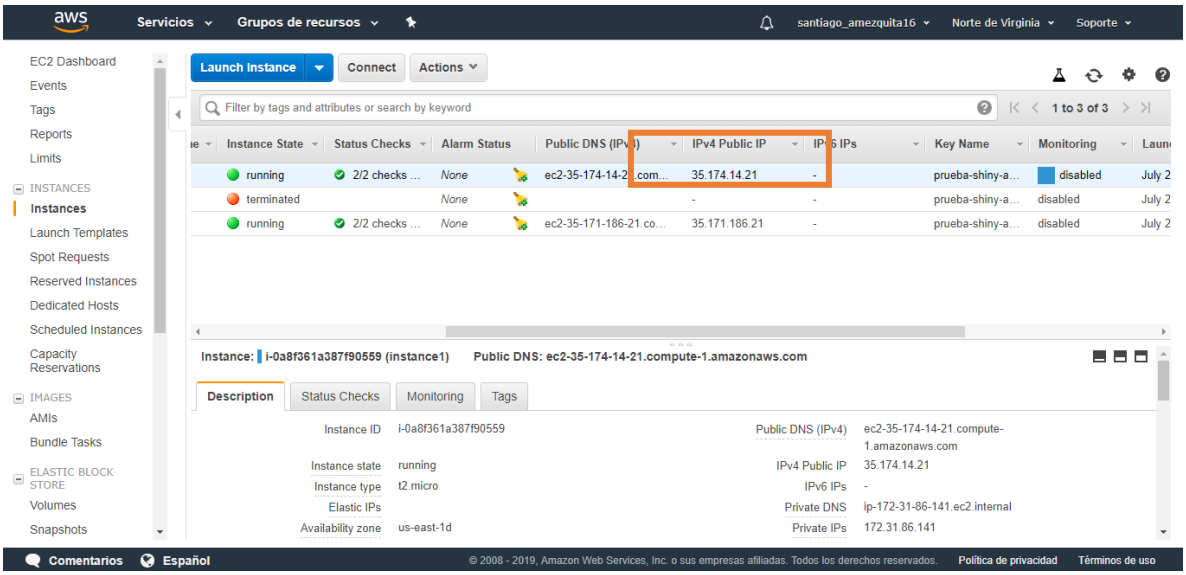
13. Download the latest shiny-server version in the bottom of this page: https://www.rstudio.com/products/shiny/download-server/

For me it is:

```
sudo apt-get install gdebi-core
wget https://download3.rstudio.org/ubuntu-14.04/x86_64/shiny-server-1.5.9.923-amd64.deb
sudo gdebi shiny-server-1.5.9.923-amd64.deb
```

14. Access the shiny app from the server

In order to access to the default shiny app that comes with the server, you have to locate the IPv4 from your AMI. This information is here:

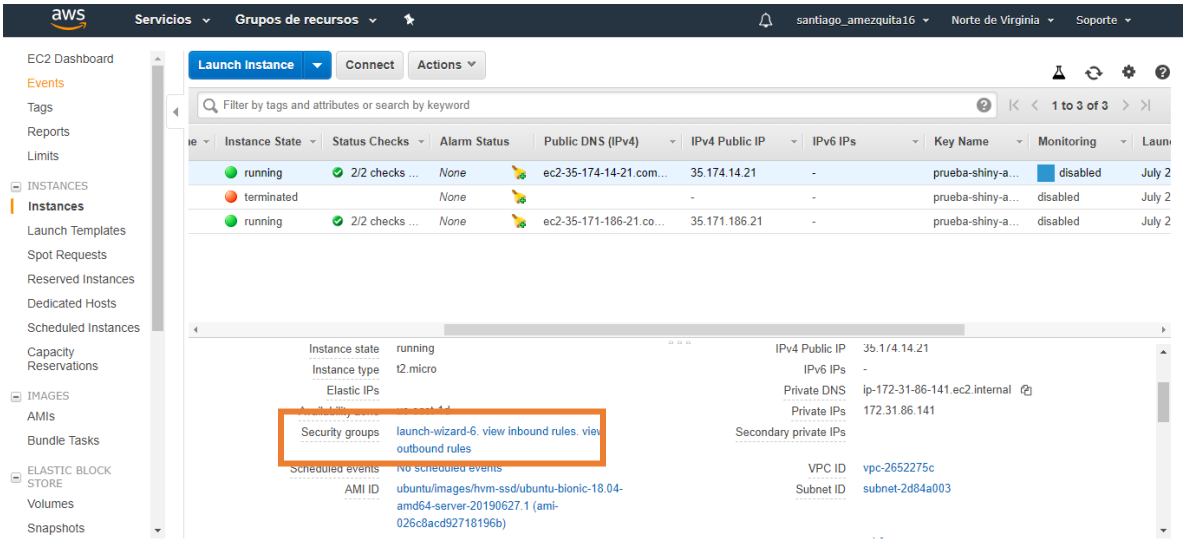


In my case it is 35.174.14.21 (This won't work because I shutted down the server after this tutorial).

If you type 35.174.14.21:3838 in your browser to access, it wouldn't work because the 3838 port isn't allowed for access.

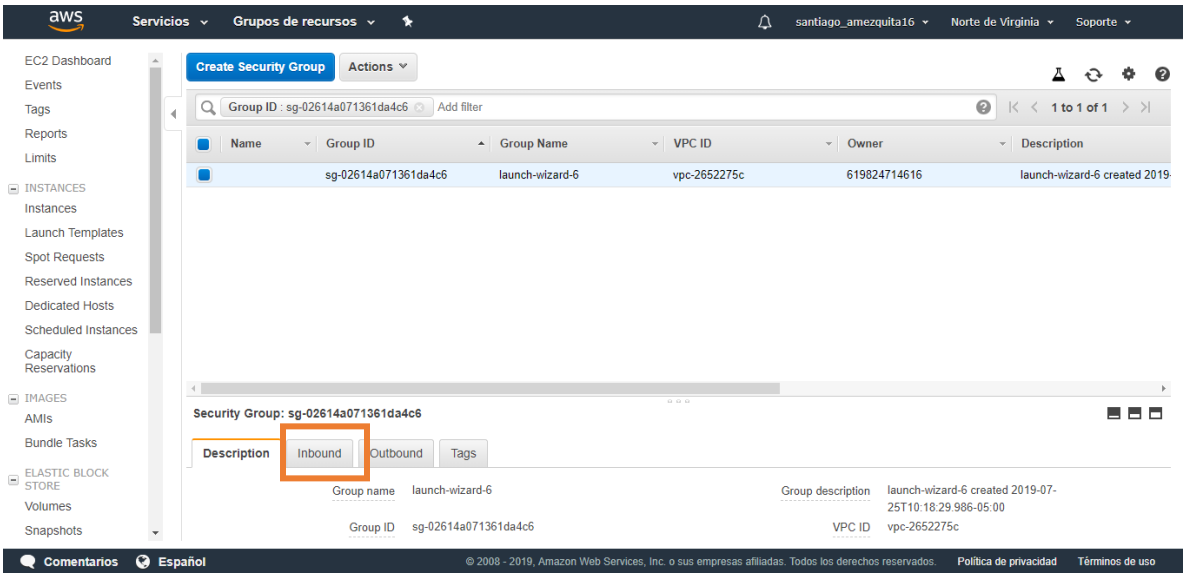
You must create a rule in the AWS server to allow the 3838 port to be accessible.

15. To allow access to the 3838 port you must select the instance and in the bottom part, there's a section called "Security groups":



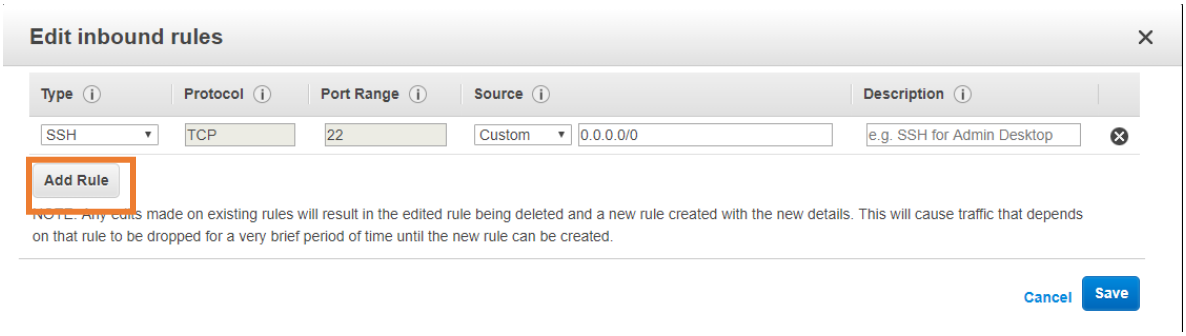
Click in launch-wizard-6

In this new page, click on "Inbound"



Click on edit

Click on "Add Rule"

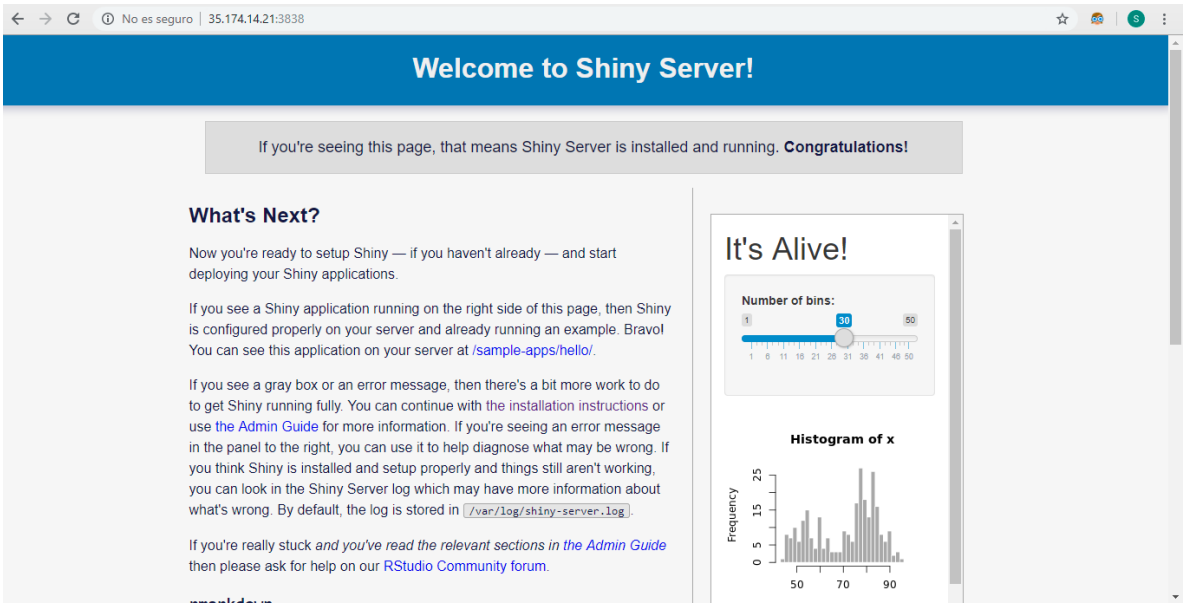


Select and add the following options to the rule:

- Type: Custom TCP Rule
- Protocol: TCP
- Port Range: 3838
- Source: Custom 0.0.0.0/0

Click on save:

Now if you tipe yorIPAddress:3838 you should see the default shiny apps on the AWS server.

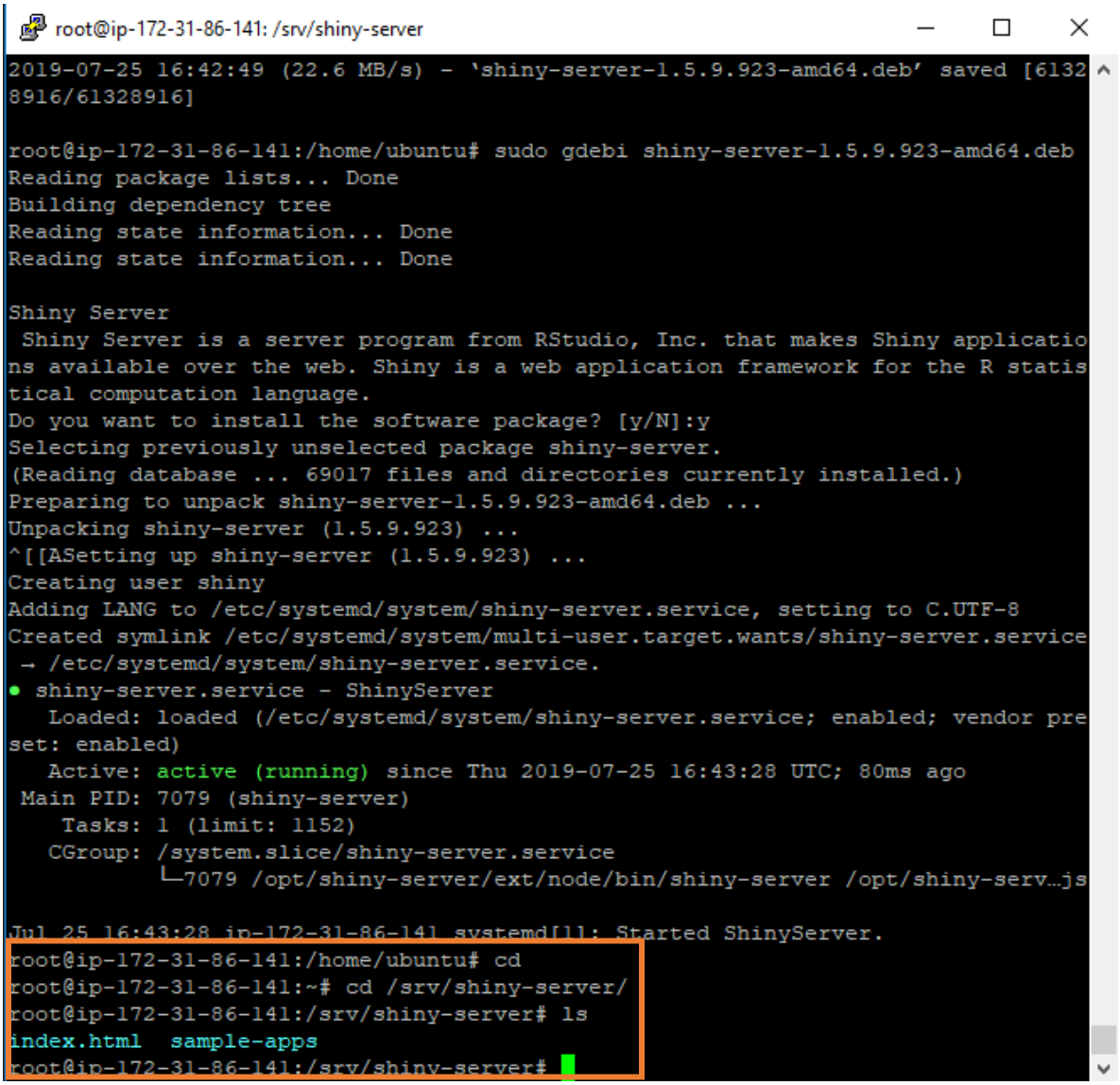


16. The information corresponding to shinyApps in the AMI is located on /srv/shiny-server. To access to this information on your AMI, you should type:

```
cd /srv/shiny-server
```

To see which elements are on that folder, type “ls”

Something like this will show up:



That means that you have two files on that folder: index.html and a folder called sample-apps. If you look deep inside the folder “sample-apps” you can see two more folders (this is done using the command line “cd sample-apps” and then “ls”):

```
root@ip-172-31-86-141:/srv/shiny-server# cd sample-apps
root@ip-172-31-86-141:/srv/shiny-server/sample-apps# ls
hello  rmd
root@ip-172-31-86-141:/srv/shiny-server/sample-apps#
```

This shows you that you have two folders, one called “hello” and other called “rmd”. Lets see what “hello” folder has with the command line “cd hello” and “ls”:

```
root@ip-172-31-86-141:/srv/shiny-server/sample-apps# cd hello/
root@ip-172-31-86-141:/srv/shiny-server/sample-apps/hello# ls
server.R  ui.R
root@ip-172-31-86-141:/srv/shiny-server/sample-apps/hello#
```

In the folder “hello” are the files server.R and ui.R, the basic structures for any shinyApp.

If you want to go back to the previous folder, you must use the command “cd ..” and if you want to go back to the root just type “cd”.

To remove a file from the AMI you can use the command:

```
sudo rm fileName
```

```
root@ip-172-31-86-141:/srv/shiny-server/sample-apps# cd ..
root@ip-172-31-86-141:/srv/shiny-server# ls
index.html  sample-apps
root@ip-172-31-86-141:/srv/shiny-server# sudo rm index.html
root@ip-172-31-86-141:/srv/shiny-server# ls
sample-apps
root@ip-172-31-86-141:/srv/shiny-server#
```

In this case, we removed index.html, now our shiny server looks like this:

Where you can access all the apps within the sample-apps folder.

You can add files from your local machine to the AMI by using WinSCP, which can connect directly with PuTTy.

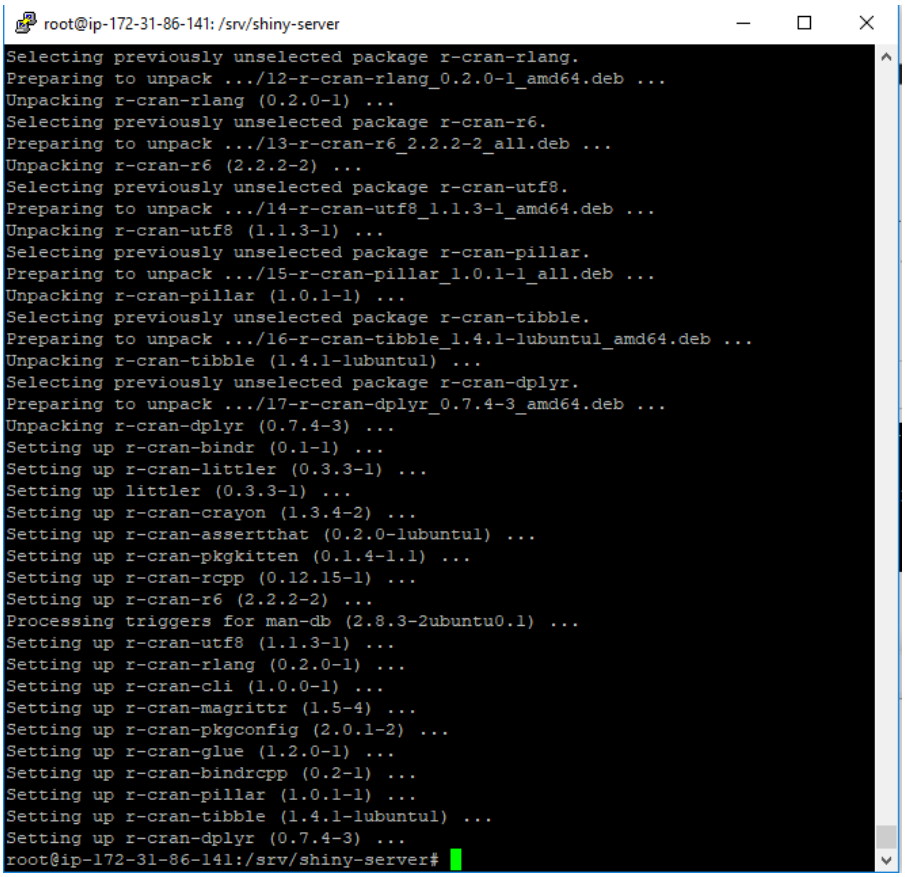
Installing R packages into your AMI

17. Installing packages on AWS machine is a fundamental part of the performance of any application. Sometimes, you must install them via R, and sometimes, via console commands.

In this example we're going to install a very important package for data science called "dplyr". In my case, when I tried to install it via R, it crashed my AMI, therefore, I used the console command to install it. My recommendation will be to install it with the console command and wait:

```
sudo apt-get install r-cran-dplyr
```

If you fail to do so via console, install the package via R with the command "install.packages("packageName")". If by some reason it fails again or takes too long, it may be due to the dependencies of the packages with other packages. You can stop the installation of the package in R by pressing Ctrl+C and it will show you which libraries are the dependencies to install the packages and try to install them using the command used above (sudo apt…).



Saving the logs of your apps in AMI

18. When an app fails, shiny has the bad habit to don't show the whole log of errors sometimes. To change this execute the following command:

```
sudo nano /etc/shiny-server/shiny-server.conf
```

In this file add the following line **AT THE TOP OF IT:**

```
preserve_logs true;
```

Then, press Ctrl+O and then Enter. Quit the file with Ctrl+X.

After this is done, execute the following command:

```
sudo systemctl reload shiny-server
```

The logs are stored on the path "/var/log/shiny-server". This feature allows you to see where your app crashes (in case it does).

Important ubuntu Console commands

- Get the root user permissions: sudo su
- Access R in ubuntu console: sudo R

- Access to a path in ubuntu console: cd /path/
- See which files are in a folder: ls
- Edit a file in the ubuntu console: nano filename.extension
- Install a package in ubuntu console: sudo apt-get install r-cran-packageName
- Reset the AWS server through ubuntu console (this should be done each time you upload a new app or apply a change on any of the R scripts): systemctl restart shiny-server