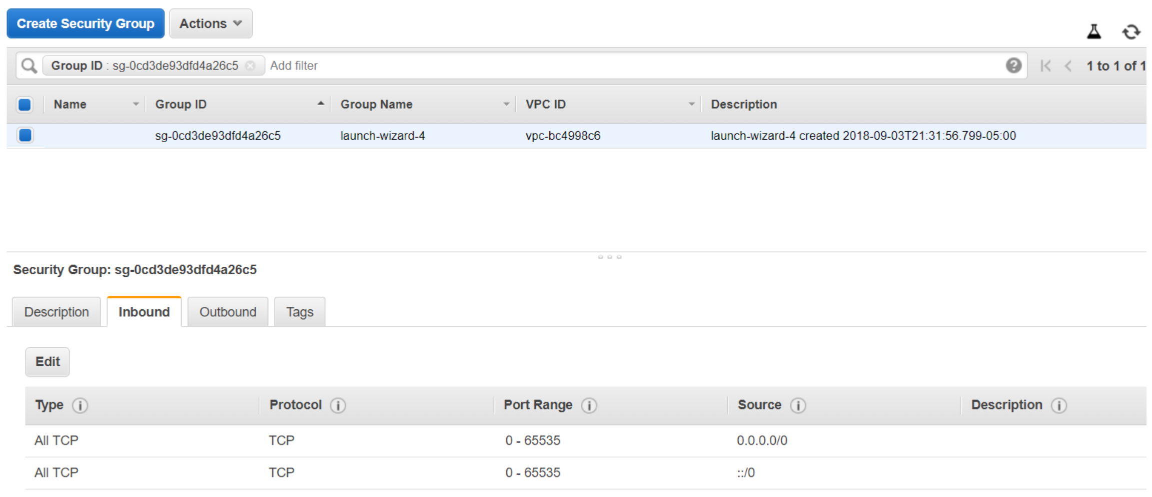
Guide to use two EC2 instances

This guide can help you in getting started with the bonus question of using multiple EC2 instances. In this guide you will be learning on using two EC2 instances for a simple pub-sub example. One instance has a ZMQ publisher which binds to a socket and starts publishing data, on the other EC2 instance is a ZMQ subscriber and it connects to a similar socket and starts receiving messages published by the publisher.

To get started you need to create two EC2 instances using this guide https://github.com/vu-resilient-distributed-18/vu-resilient-distributed-18.github.io/blob/master/aws-setup/AWS-guide.pdf

Only thing you have to keep in mind is to configure the security groups. The rules you setup should allow using the TCP ports you want to use (or all TCP ports) from anywhere. The inbound and outbound of your instance should look the figures below.

Inbound rules to allow access of all the TCP ports from anywhere

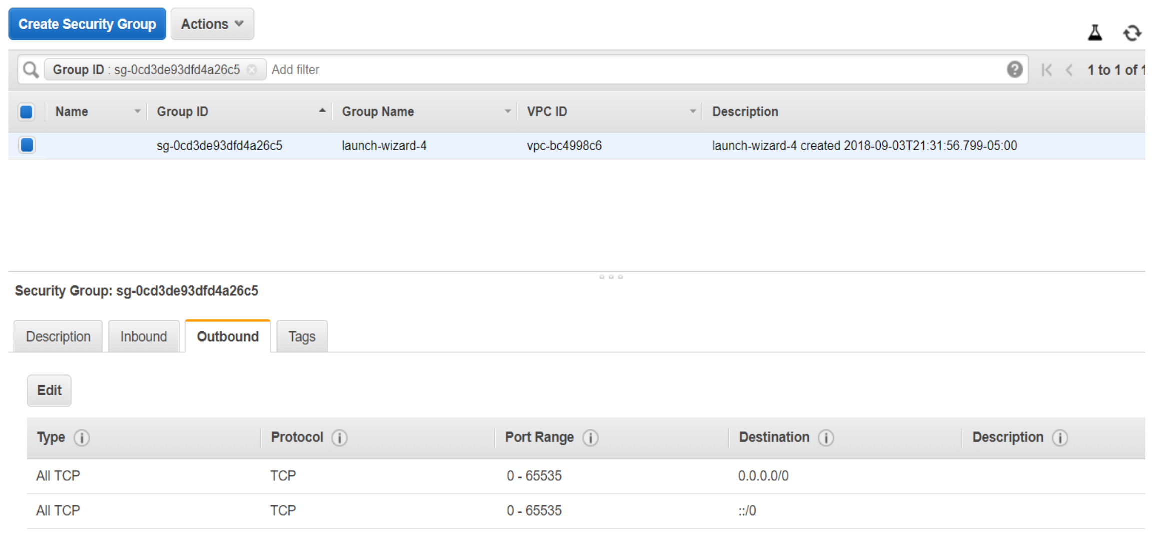


Name	Group ID	Group Name	VPC ID	Description
	sg-0cd3de93dfd4a26c5	launch-wizard-4	vpc-bc4998c6	launch-wizard-4 created 2018-09-03T21:31:56.799-05:00

Security Group: sg-0cd3de93dfd4a26c5

Type	Protocol	Port Range	Source	Description
All TCP	TCP	0 - 65535	0.0.0.0/0	
All TCP	TCP	0 - 65535	::/0	

Outbound rules to allow access of all the TCP ports from anywhere



Name	Group ID	Group Name	VPC ID	Description
	sg-0cd3de93dfd4a26c5	launch-wizard-4	vpc-bc4998c6	launch-wizard-4 created 2018-09-03T21:31:56.799-05:00

Security Group: sg-0cd3de93dfd4a26c5

Type	Protocol	Port Range	Destination	Description
All TCP	TCP	0 - 65535	0.0.0.0/0	
All TCP	TCP	0 - 65535	::/0	

Once you have the two instance setup along with the required security group, you are ready to use the instances for your assignment.

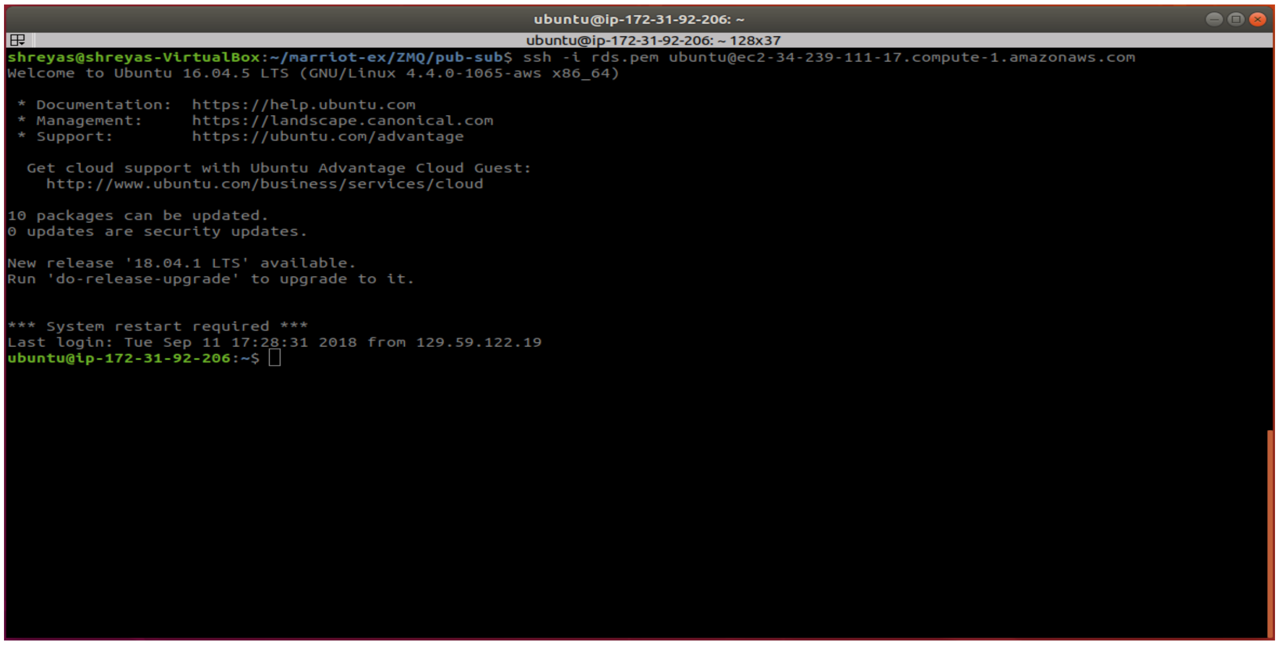
SSH into the EC2 instances

In the previous tutorial you guys have probably used putty to login to your instances. This is one way of getting into your instance, however if you guys are comfortable using ubuntu, then there is an easy method of logging into your instance using SSH. You could read through this document https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/AccessingInstancesLinux.html to get started with SSH.

Command: **ssh –i rds.pem ubuntu@ec2-34-239-111-17.compute-1.amazonaws.com**

Here you are logging into your instance using the identity file (private key). Very important thing to note is the file permission for the private key. You need to provide a only a read permission to the user using the command given below. If you do not set this permission then you cannot login.

File permission for private key: **chmod 400 key.pem (this is very important)**



```
shreyas@shreyas-VirtualBox:~/marriot-ex/ZMQ/pub-sub$ ssh -i rds.pem ubuntu@ec2-34-239-111-17.compute-1.amazonaws.com
Welcome to Ubuntu 16.04.5 LTS (GNU/Linux 4.4.0-1065-aws x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

  Get cloud support with Ubuntu Advantage Cloud Guest:
    http://www.ubuntu.com/business/services/cloud

10 packages can be updated.
0 updates are security updates.

New release '18.04.1 LTS' available.
Run 'do-release-upgrade' to upgrade to it.


*** System restart required ***
Last login: Tue Sep 11 17:28:31 2018 from 129.59.122.19
ubuntu@ip-172-31-92-206:~$
```

So, if you have followed all of the steps correctly you will be able to login to your instance. The login looks like the one shown in the figure above.

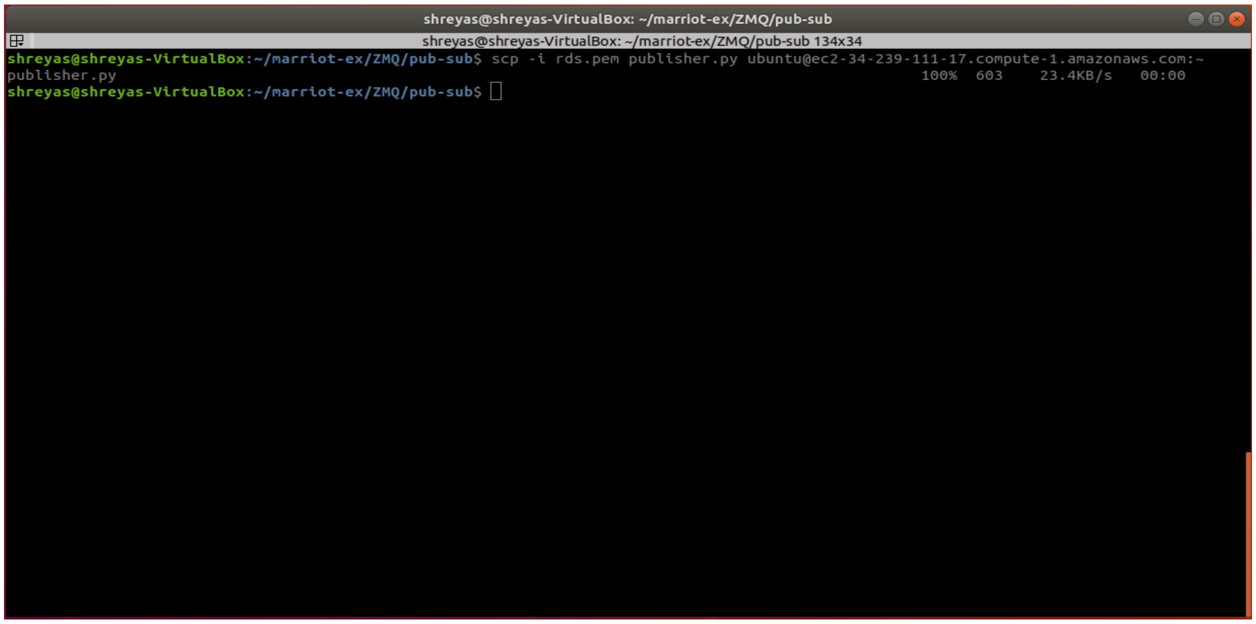
Copying Files to EC2 instance

Once you have successfully logged in to your instance, you will have to transfer or copy your files onto your instance. To do so you could use the command given below.

Copying using scp: **scp –i file.py ubuntu@ ec2-34-239-111-17.compute-1.amazonaws.com:~**

This would copy the file.py to the EC2 instance.

The copy using scp will look as shown in the figure below.


shreyas@shreyas-VirtualBox: ~/marriot-ex/ZMQ/pub-sub
shreyas@shreyas-VirtualBox: ~/marriot-ex/ZMQ/pub-sub 134x34

```
shreyas@shreyas-VirtualBox:~/marriot-ex/ZMQ/pub-sub$ scp -i rds.pem publisher.py ubuntu@ec2-34-239-111-17.compute-1.amazonaws.com:~
publisher.py                                                                                              100%  603    23.4KB/s   00:00
shreyas@shreyas-VirtualBox:~/marriot-ex/ZMQ/pub-sub$ 
```

You guys are all set to work with the bonus part of the assignment.

Pub-Sub example

Let us look at the same pub-sub example in the repository https://github.com/vu-resilient-distributed-systems/lectures-fall-2019/tree/master/Module-1-Networking/examples/ZMQ/pub-sub and let us run each of the scripts in a different instance.

Publisher.py

```
def main():

    context = zmq.Context()
    socket= context.socket(zmq.PUB)
    socket.bind("tcp://*:5200")#PUB has to bind to an ip address.
    #Here we use a * aka wild card, which opens up all the interfaces
    #This is an important change you need to do when you run the publisher on EC2
    while True:
        for i in range (0,100):
            msg = randint(1,100)#publishing random numbers every 2s
            socket.send(str(msg))#Send messages
            print(msg)
            time.sleep(2)
    socket.close()

if __name__=="__main__":
    main()
```

As seen in the code above, we can see the only change we need to make is the ip address. So, if you want to run the publisher on the instance, then you use a * for the ip address. This means that you are attaching your sockets to all the available interfaces on your EC2 instance. (You can find all the available network interfaces using command **ifconfig**).

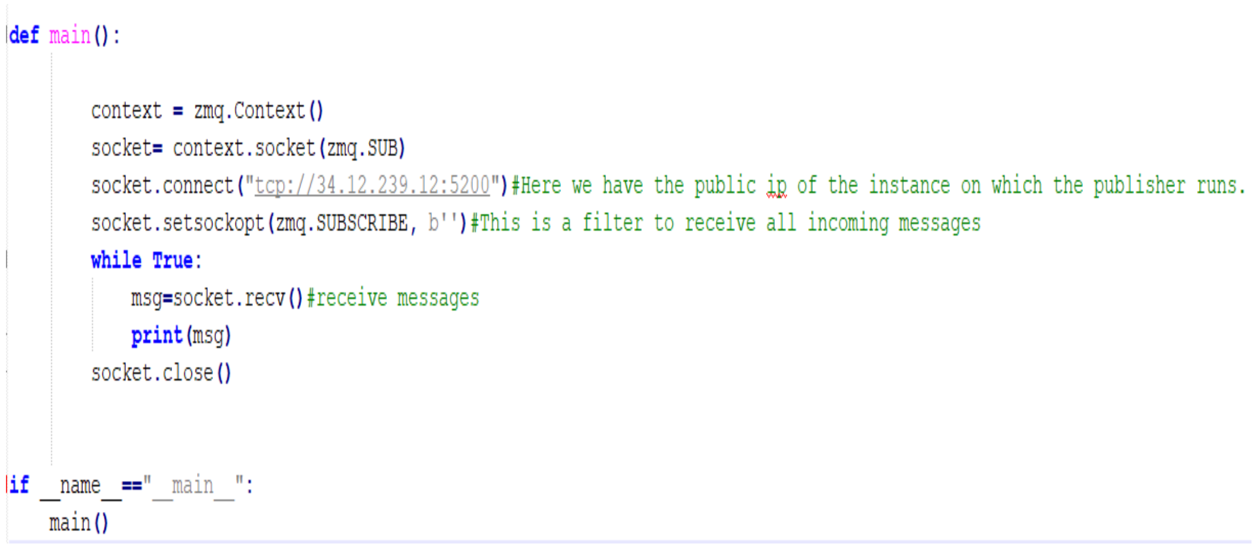
Ref: **http://api.zeromq.org/master:zmq-tcp**

Note: If you do not want to use the *, then you can use the private ip address of the instance, however this is not encouraged)

Subscriber.py

The subscriber.py remains the same, however now the subscriber connects to the public ip of the instance running the publisher.py.

As seen in the figure, the socket.connect(tcp://**34.12.239.12**:5200) uses the public ip of the instance running the publisher.py.



```
def main():

    context = zmq.Context()
    socket= context.socket(zmq.SUB)
    socket.connect("tcp://34.12.239.12:5200")#Here we have the public ip of the instance on which the publisher runs.
    socket.setsockopt(zmq.SUBSCRIBE, b'')#This is a filter to receive all incoming messages
    while True:
        msg=socket.recv()#receive messages
        print(msg)
    socket.close()


if __name__=="__main__":
    main()
```

So with these changes to your code, you could execute the pub-sub example on two different EC2 instances.