6/8/2011

Distributed convex Belief Propagation

Amazon EC2 Tutorial

Alexander G. Schwing, Tamir Hazan, Marc Pollefeys and Raquel Urtasun

Distributed convex Belief Propagation

Amazon EC2 Tutorial

Introduction

This document briefly describes the required steps to run the distributed convex Belief Propagation algorithm on an Amazon EC2 cluster. We assume an Amazon Web Services (AWS) account. Further we recommend the following two links:

- http://aws.amazon.com/about-aws/build-a-cluster-in-under-10/
- http://docs.amazonwebservices.com/AmazonEC2/gsg/2007-01-19/putty.html

Note that you can apply for free AWS credits using the first link.

Creating a reference machine

We first sign into the AWS management console using our AWS account by browsing to aws.amazon.com.

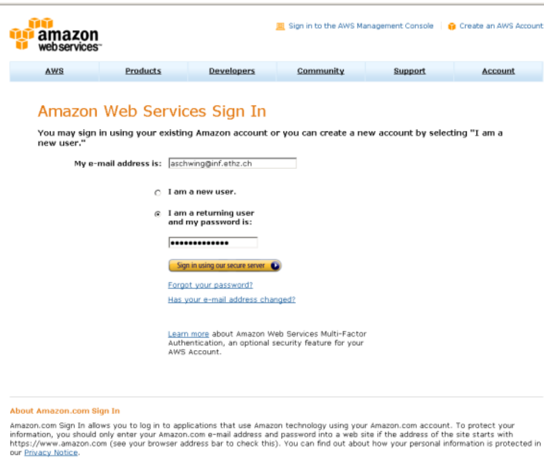

Figure 1: Sign in using AWS account

After changing to the EC2 Tab the screen looks similar to Figure 2, where we press the “Launch Instance” button.

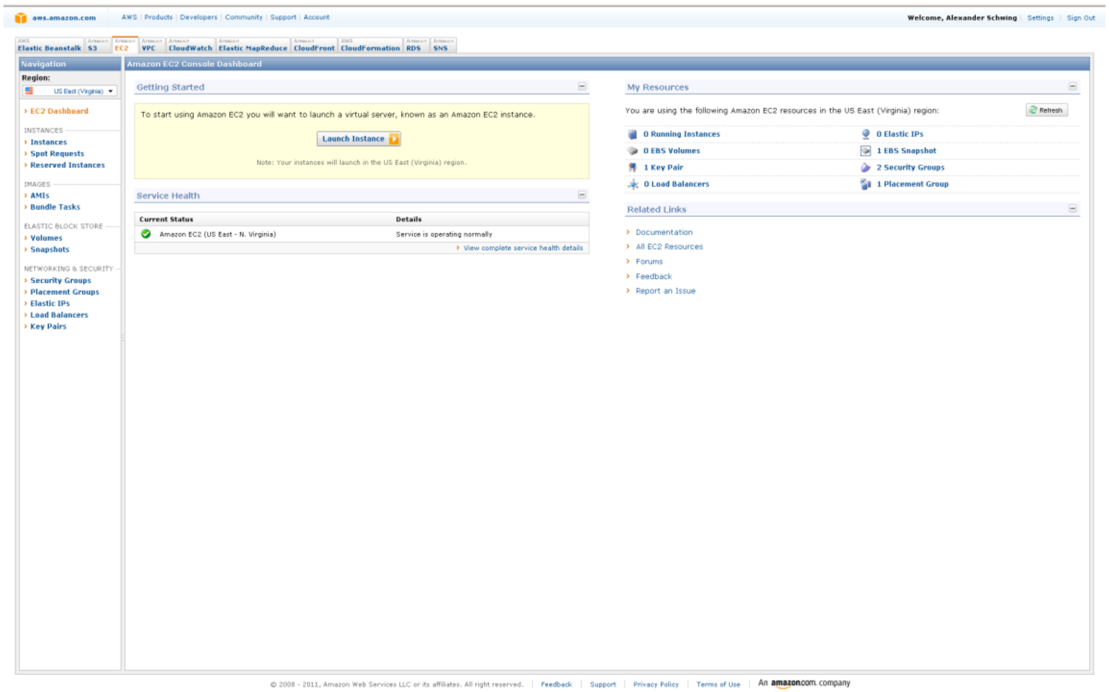

Figure 2: After logging in to AWS and changing to the Ec2 tab page.

A window appears where we choose the Community AMIs (Amazon Machine Image) tab, type "hvm" into the text box to restrict the search range and sort according to platform such that Linux based systems are shown first. We obtain a screen looking similar to Figure 3. Here we select the image named "amazon/EC2 CentOS 5.4 HVM AMI" which is based on Cent OS.

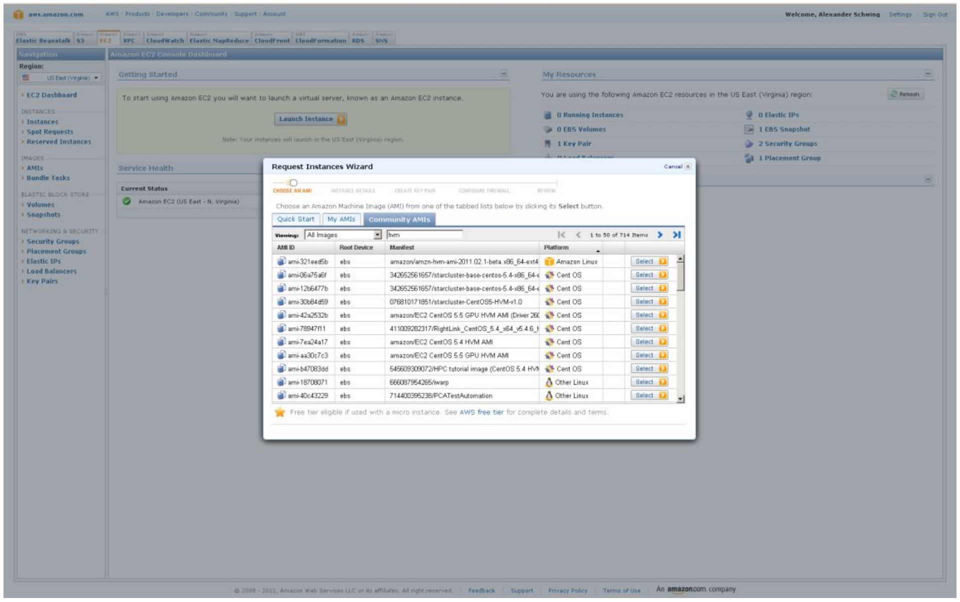

Figure 3: Choosing an Amazon machine image.

Next we specify the machine settings such as type (Figure 4) "cc1.4xlarge", placement group which we named "cluster" (Figure 5). The keys (Figure 6) can be left empty.

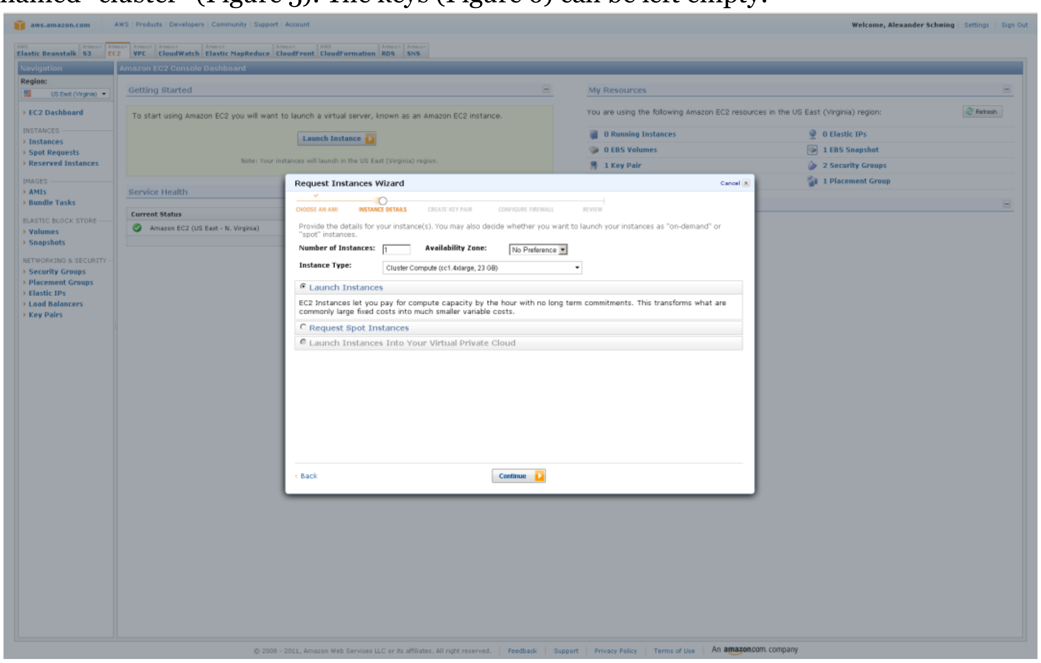

Figure 4: The Machine type.

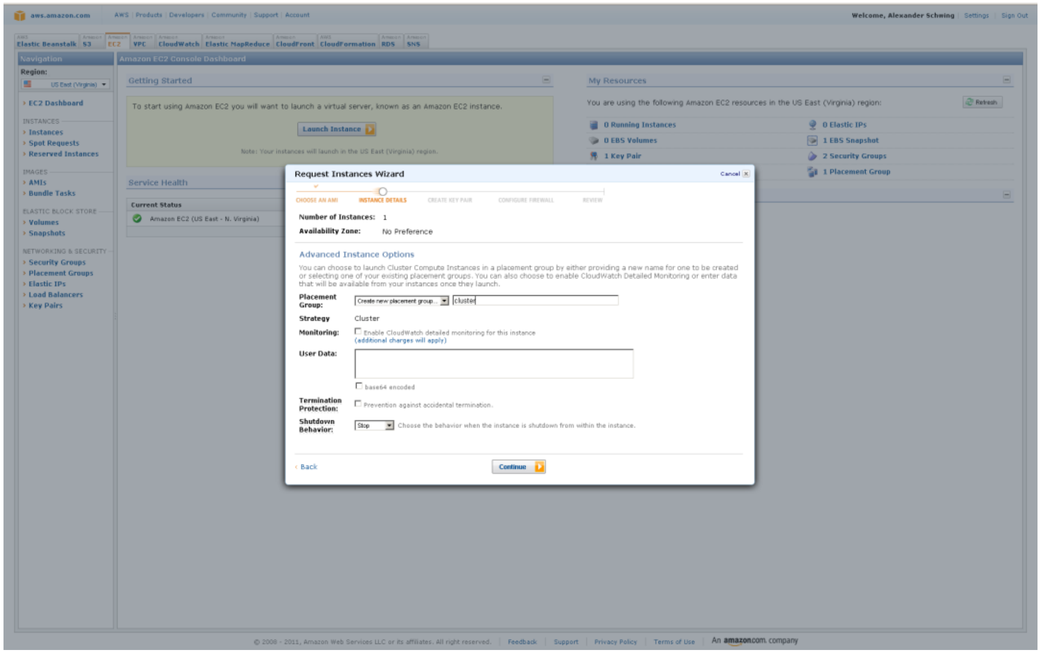

Figure 5: The placement group.

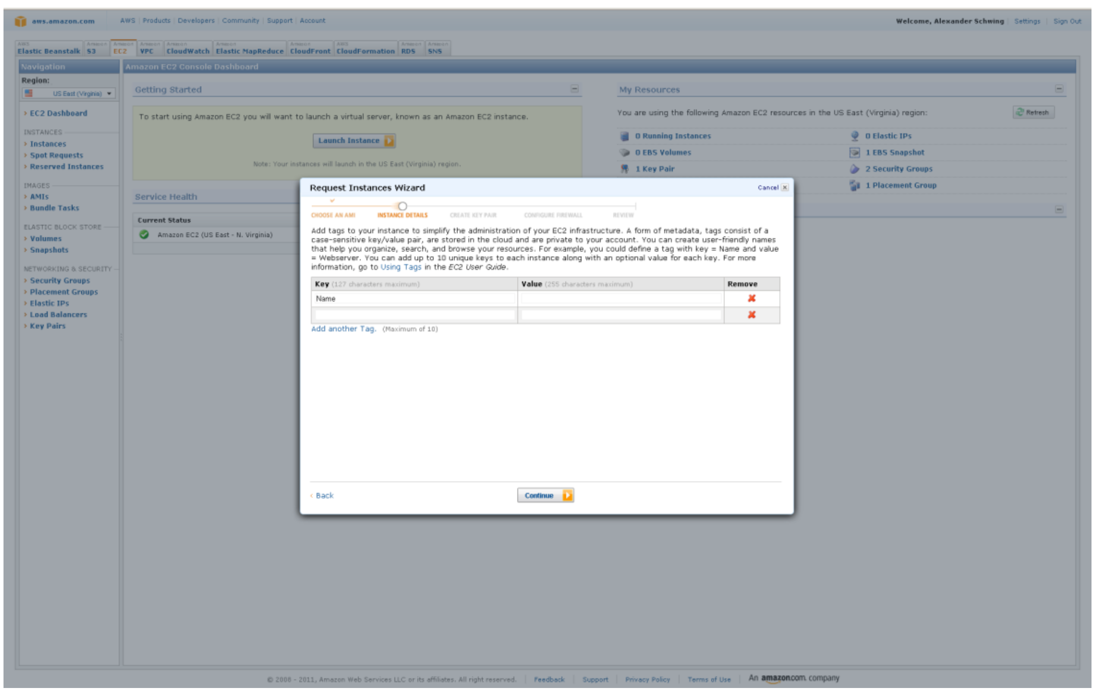

Figure 6: Key and value pairs.

Next we need to create a login key. We specify the name "amazon-hpc-1" and download the respective file named "amazon-hpc-1.pem" as indicated in Figure 7.

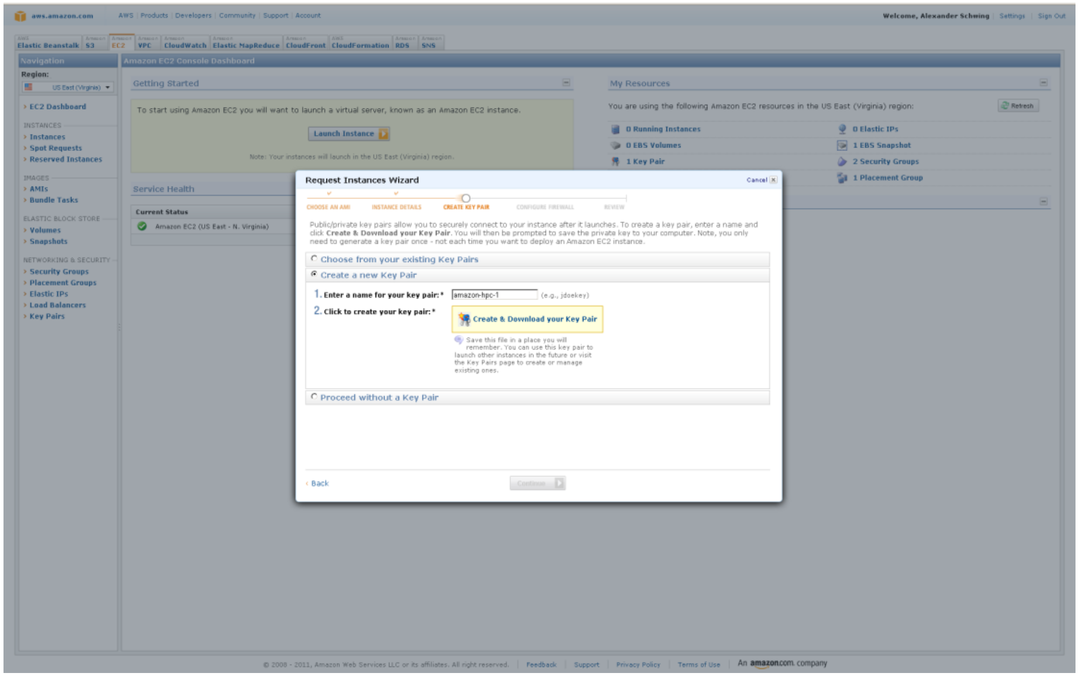

Figure 7: Generating and downloading the login key.

Afterwards, we have to change the firewall settings such that we can login after all. This is done on the next screen, which looks similar to Figure 8. We create a new security group named "hpc" and allow inbound traffic from any source on port 22 (SSH).

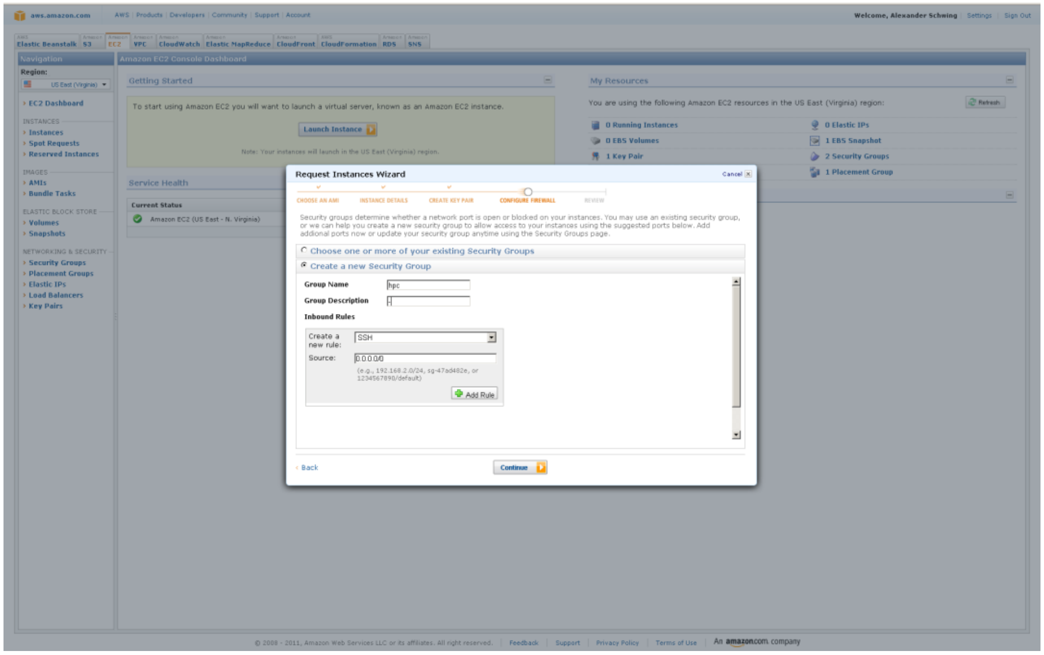

Figure 8: The firewall settings.

We finally review the chosen settings (Figure 9) before launching the machine.

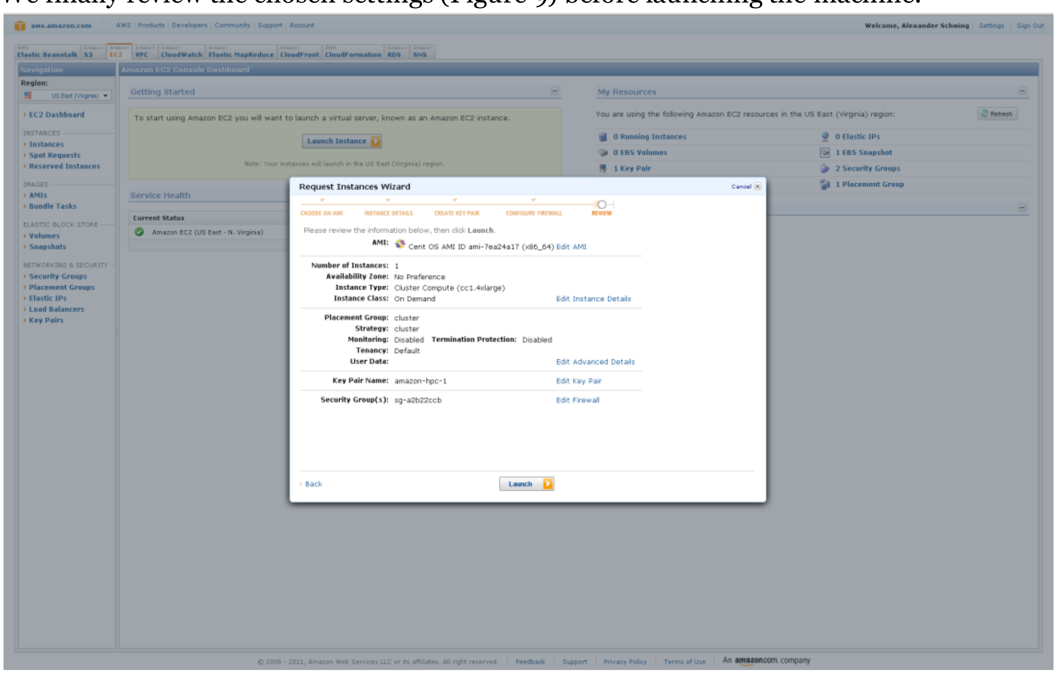

Figure 9: Reviewing the settings.

It will take a while till the requested machine is available and accessible. If working on a Windows computer we can meanwhile convert the obtain key file (amazon-hpc-1.pem) to a PuTTY compatible version by following the steps described in http://docs.amazonwebservices.com/AmazonEC2/gsg/2007-01-19/putty.html. Essentially we load the *.pem file with PuTTYgen (File -> Load private key) and save it again as amazon-hpc-1.ppk via File -> Save private key. To efficiently log into the machines we use the PuTTY authentication agent (Pageant) by pressing the "Add Key" button, choosing the converted *.ppk file and hitting close.

Now we are ready to log into the machine provided by Amazon. To this end we use ssh or a compatible client, e.g. PuTTY. For PuTTY we allow agent forwarding as illustrated in Figure 10.

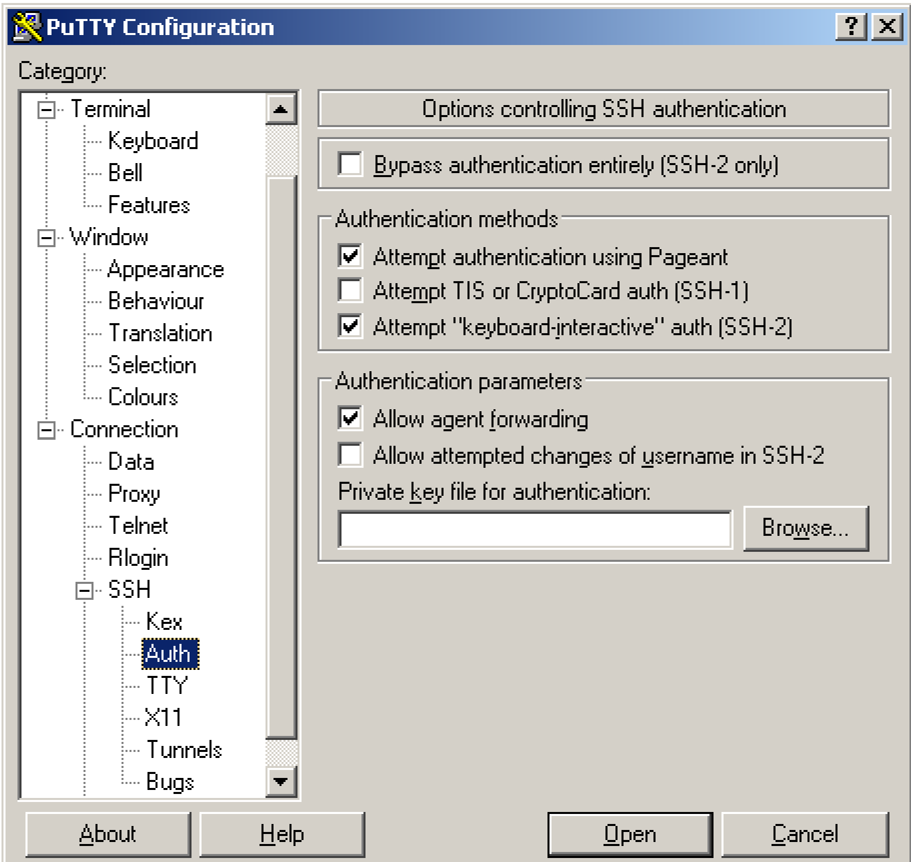

Figure 10: Allowing agent forwarding in PuTTY.

Once the machine is available (to be able to log in might take a while) we can log in by using the Public DNS name displayed on the Instances page of the EC2 tab as illustrated in Figure 11.

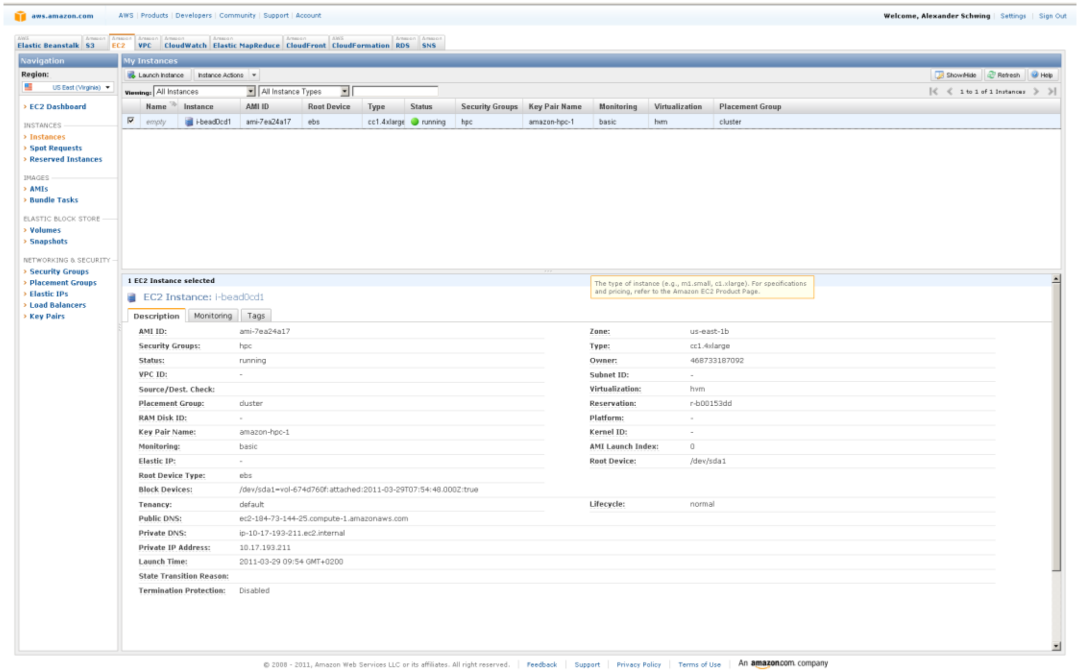

Figure 11: The machine and its Public DNS name.

We paste the name into the ssh client as illustrated in Figure 12 and hit the Open button.

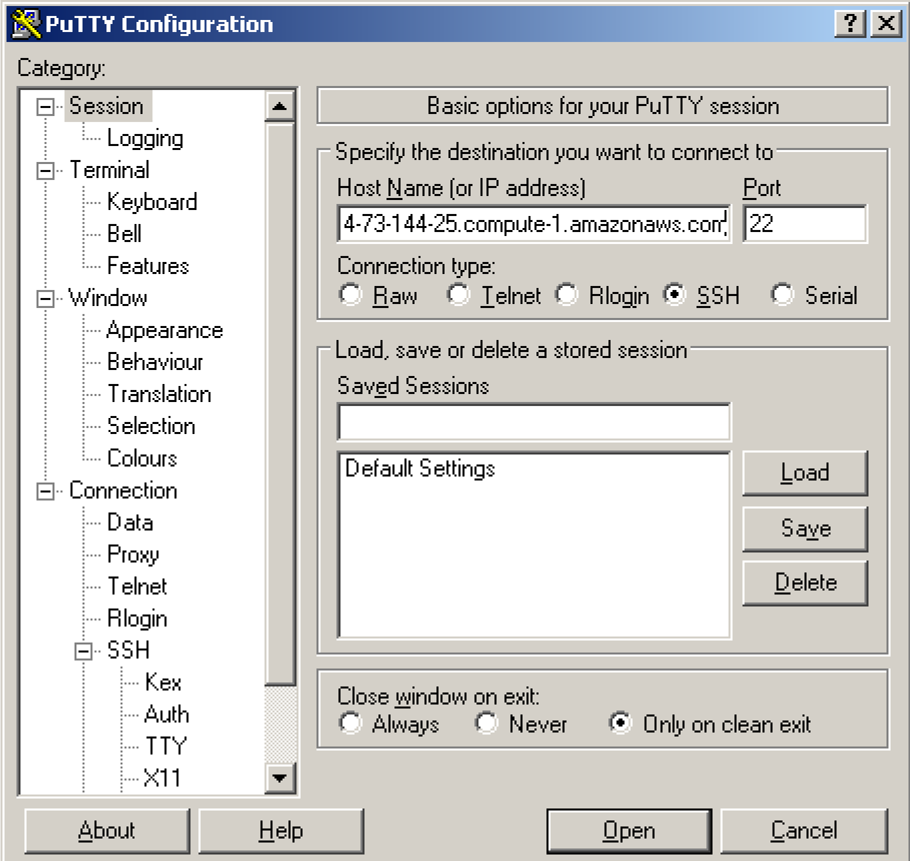

Figure 12: PuTTY with the Public DNS name.

On a Unix like operating system we log in via the following commands using the appropriate DNS name:

- chmod 400 amazon-hpc-1.pem
- ssh -a -i amazon-hpc-1.pem root@ec2-184-73-144-25.compute-1.amazonaws.com

Logged in as root, we should obtain a console similar to Figure 13.

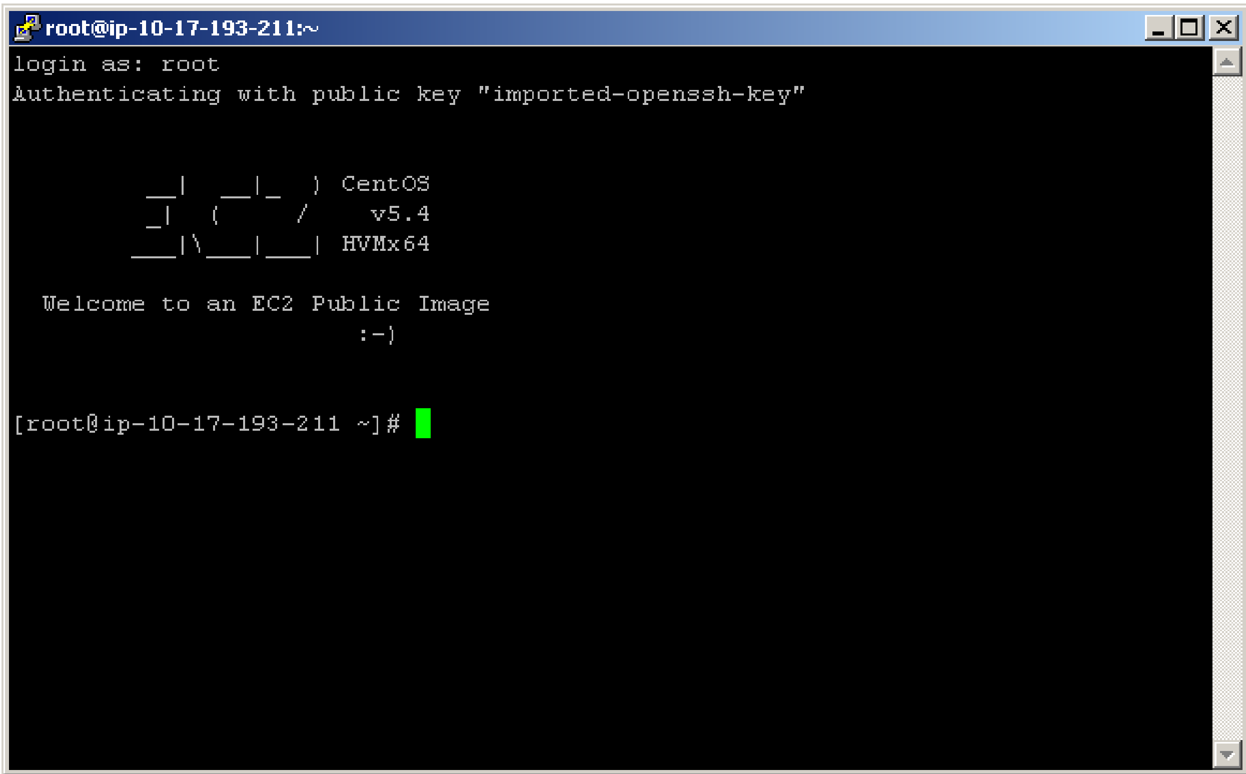

Figure 13: The console of our Amazon EC2 machine.

Setting up the reference machine

First we need to install three required packages. This is done via the following three commands:

- yum install openmpi
- yum install openmpi-devel
- yum install gcc-c++

Afterwards we adjust the path. We first find the folder where the binary files are installed using

- rpm -ql openmpi-devel | grep bin

and make sure to add the 64bit folder to the path via (in our case):

- PATH=/usr/lib64/openmpi/1.4-gcc/bin:$PATH

Next we copy the files using SCP (PSCP on Windows) from our local machine. The respective commands read as follows:

- pscp -i amazon-hpc-1.ppk dcBP.zip root@ec2-184-73-144-25.compute-1.amazonaws.com:
- scp -i amazon-hpc-1.pem dcBP.zip root@ec2-184-73-144-25.compute-1.amazonaws.com:

We also need to copy the key file which follows above command and (on a linux machine) reads as

- scp -i amazon-hpc-1.pem amazon-hpc-1.pem root@ec2-184-73-144-25.compute-1.amazonaws.com:.ssh

while making sure that the part after the @ is the Public DNS name of our machine and the files are in the same folder as our current working directory. Otherwise the paths need to be modified accordingly. After a successful transfer we unpack the archive and compile the algorithm on the Amazon machine using

- unzip dcBP.zip
- cd dcBP
- make

Finally we need to modify the SSH settings of the Amazon machine to allow for agent forwarding. Therefore, we run the following commands:

- cd ~
- chmod 400 .ssh/amazon-hpc-1.pem
- cat << EOF > .ssh/config
  ForwardAgent yes
  IdentityFile ~/.ssh/amazon-hpc-1.pem
  EOF

To check that we can login to the Amazon machine from the machine itself without any password, the following should work:

- ssh localhost

Now we are done with our reference machine and can log out.

Setting up the Cluster

To start the multiple machines of our cluster we return to the AWS management console, choose the currently running instance and create an image as illustrated in Figure 14.

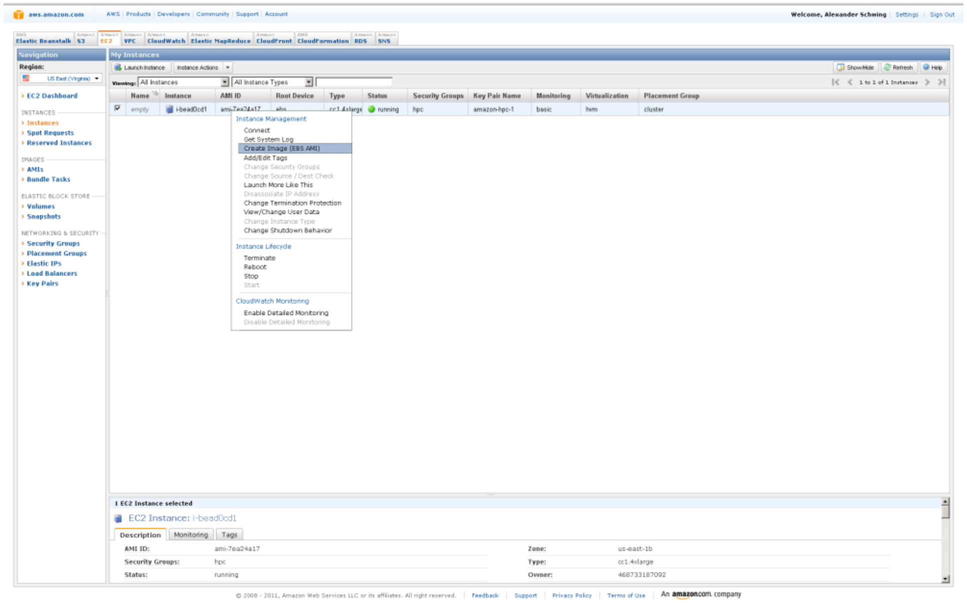

Figure 14: Creating a machine image.

We give the image a descriptive name as depicted in Figure 15 and start the process.

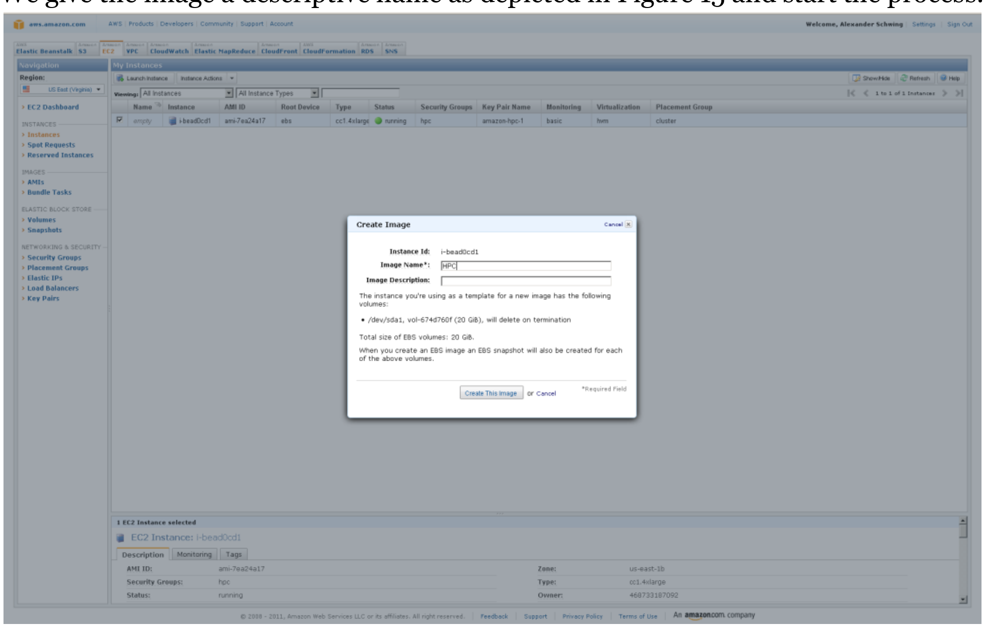

Figure 15: Choosing a name for the image.

This process can take a while and we can observe the status at the AMIs page of the EC2 tab (accessible through the navigation area on the left hand side) as illustrated in Figure 16.

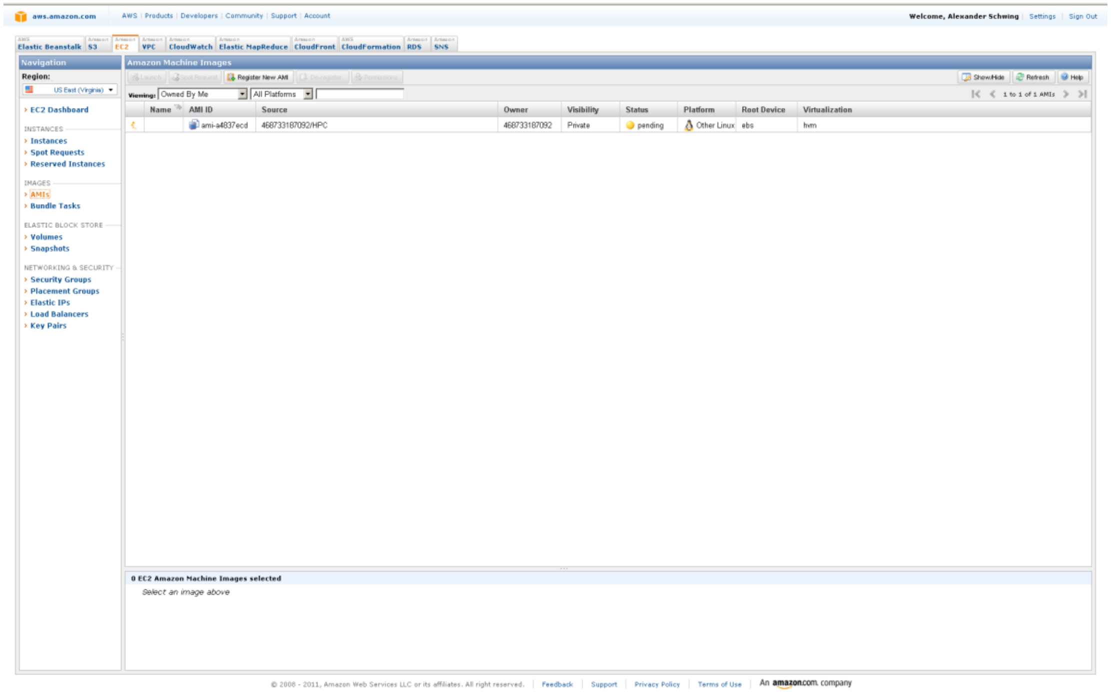

Figure 16: The status of the machine image.

After its completion we can launch an exact replica of the first machine by right-clicking the machine image and hitting launch as illustrated in Figure 17.

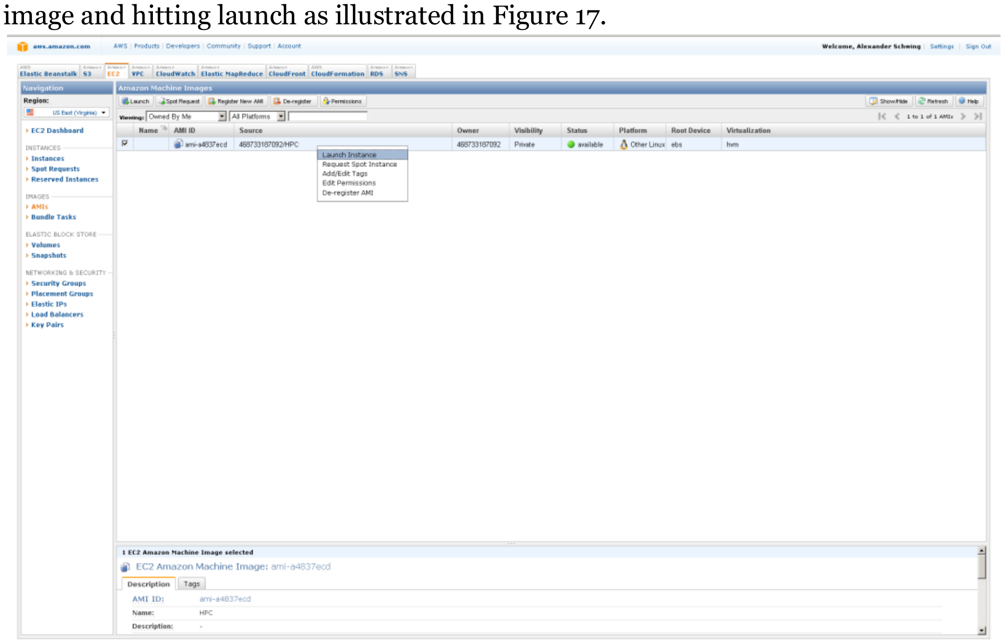

Figure 17: Starting another machine.

We choose the number of additional machines we want to start (one in our case), the correct placement group (Figure 18), the respective ssh key pair (Figure 19), and the appropriate Firewall settings (Figure 20) where we choose the previously created security group named "hpc."

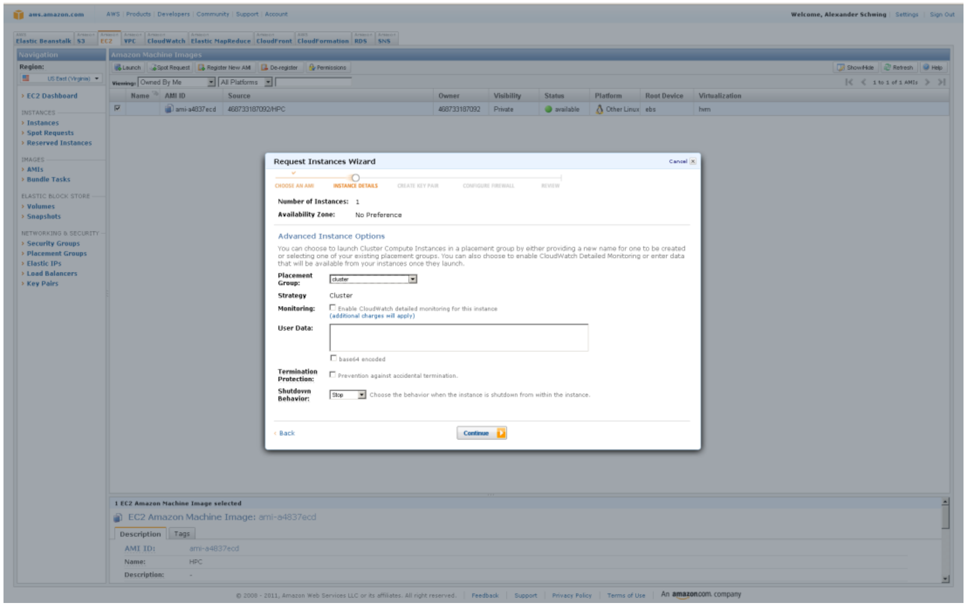

Figure 18: The placement group.

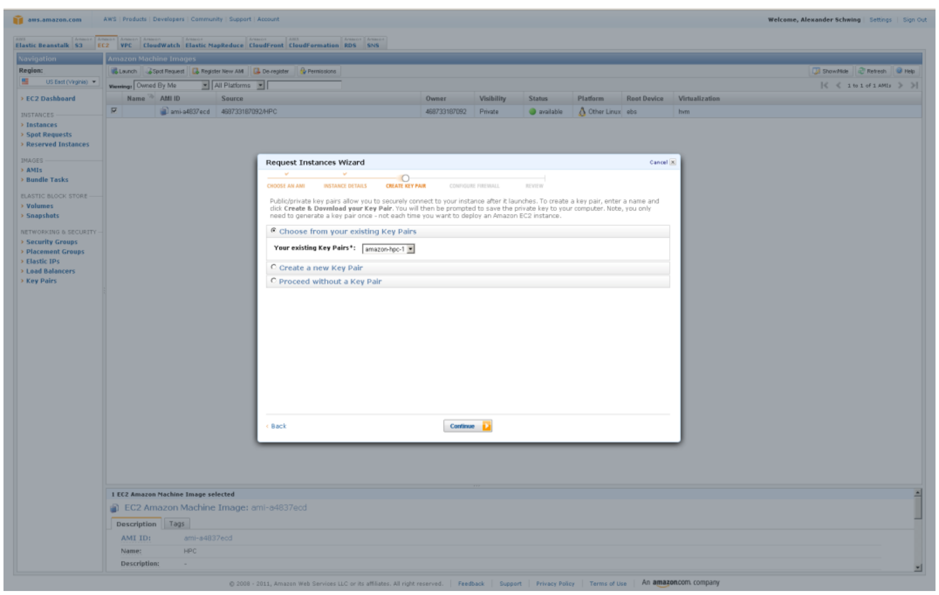

Figure 19: The ssh key pair.

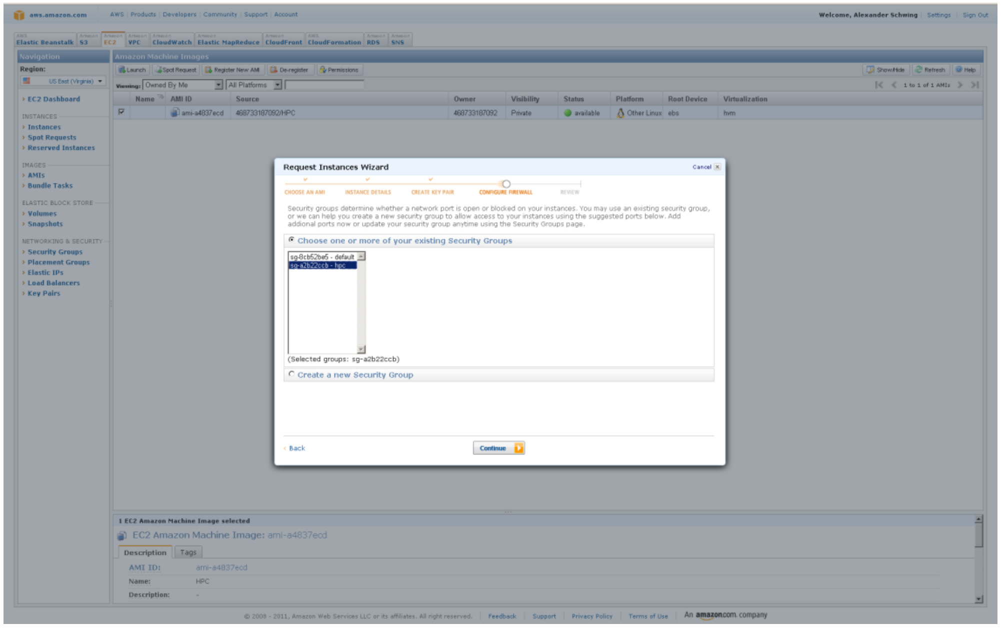

Figure 20: The firewall settings.

After a while we should have two machines running as illustrated on the instances page of the EC2 tab (see Figure 21).

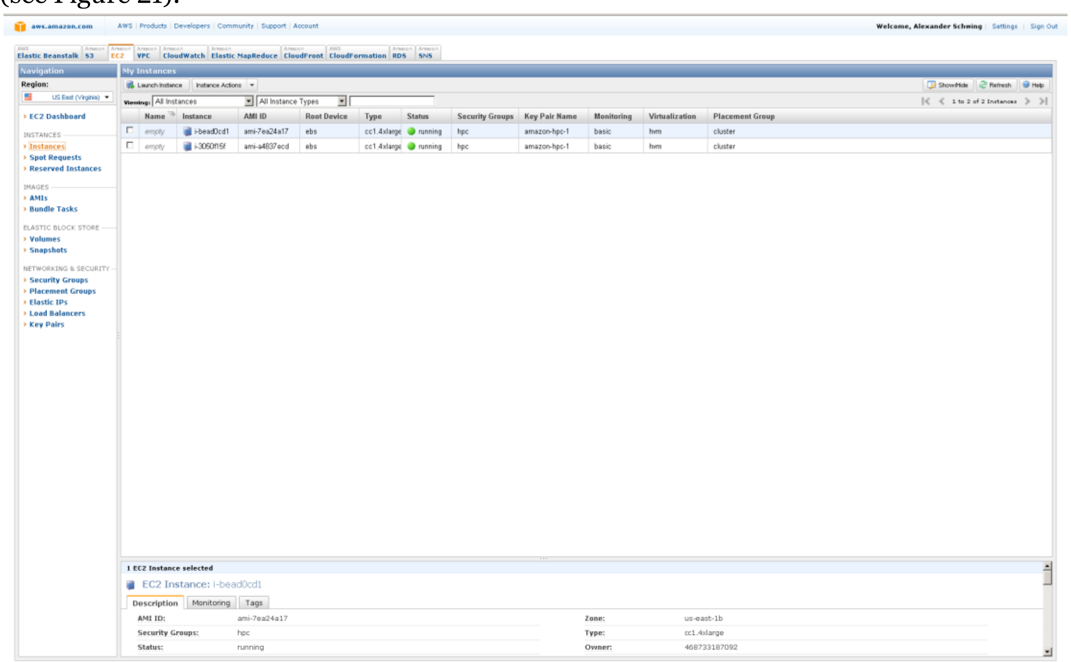

Figure 21: Two running machines.

Before logging in we still need to make sure that the members of our cluster can communicate with each other. To this end we have to modify the settings of our “hpc” security group. We choose Security Groups in the Navigation bar on the left hand side and mark the “hpc” group. On the Inbound tab at the bottom of the page we create three new rules by selecting one after the other “All TCP,” “All UDP,” and “All ICMP.” As Source we specify “hpc” each time, and click the Add Rule button. We finally apply the rule changes by hitting the respective button. The modified “hpc” security group should look similar to the illustration in Figure 22.

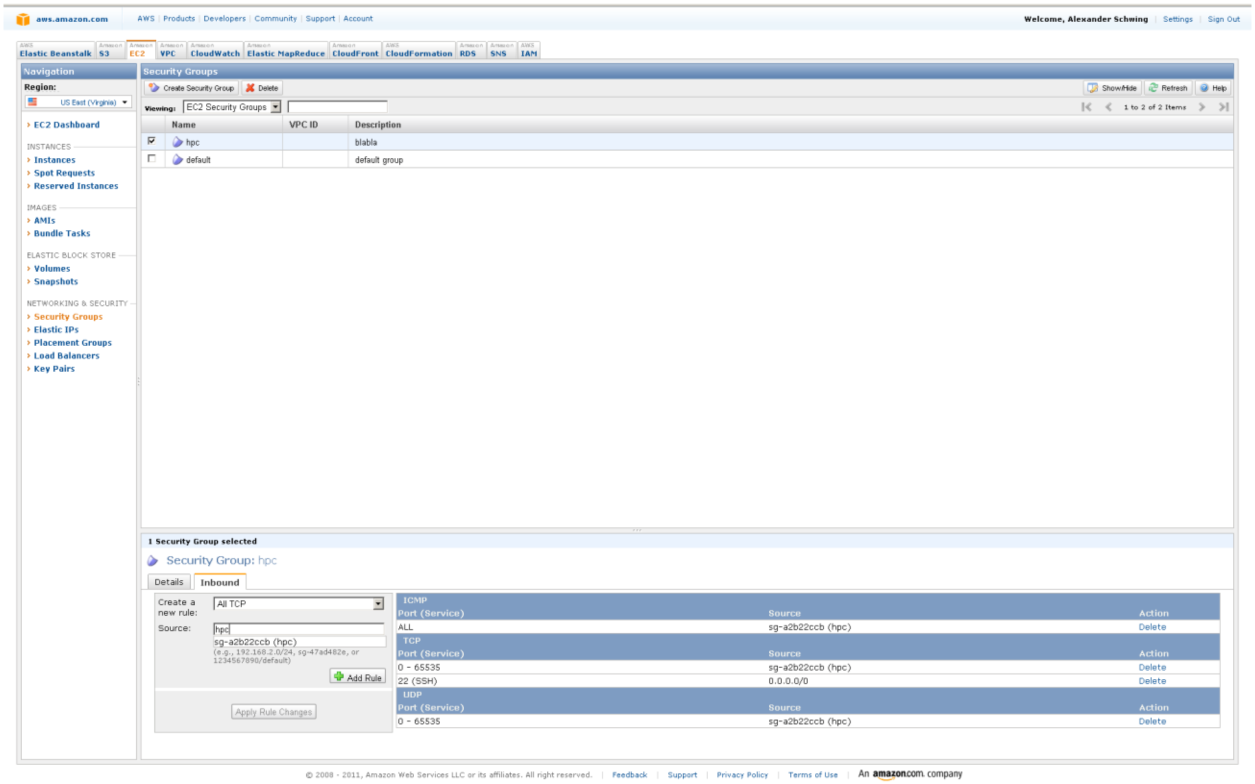

Figure 22: The modified security group.

Running a task

On Unix like operating systems we log in and start the ssh-agent via the following two commands:

- ssh -a -i amazon-hpc-1.pem root@ec2-184-73-144-25.compute-1.amazonaws.com
- eval `ssh-agent | grep -v echo`

On Windows machines we should be able to log in using PuTTY as before.

Before distributing a task onto multiple machines we have to specify the machines participating in the computation. In case of MPI, this is done via a machinefile. Hence we create a machinefile using our favorite editor (mcedit, vi, …). This file contains in every line one Private IP Address or Private DNS found in the description of the EC2 instances.

Finally we can run the distributed task using e.g.

- mpiexec -machinefile /root/machinefile -n 3 /root/dcBP/dcBP -f /root/dcBP/tsukuba.cbp -e 0 -s 10 -c 10

and copy the result using scp, e.g.

- scp LocalBeliefs.txt user@LocalMachineName:

Note that we specify the absolute paths when running the task.

Also keep in mind that for optimal performance the number of OpenMP threads should be restricted if hyperthreading is enabled (which was the case for us when using an Amazon client). To this end you have to uncomment and modify the `omp_set_num_threads(8)` command in the constructor of the Client implementation, `Client<T>::Client(…)` and provide the actual number of cores, i.e. 8 in our case.