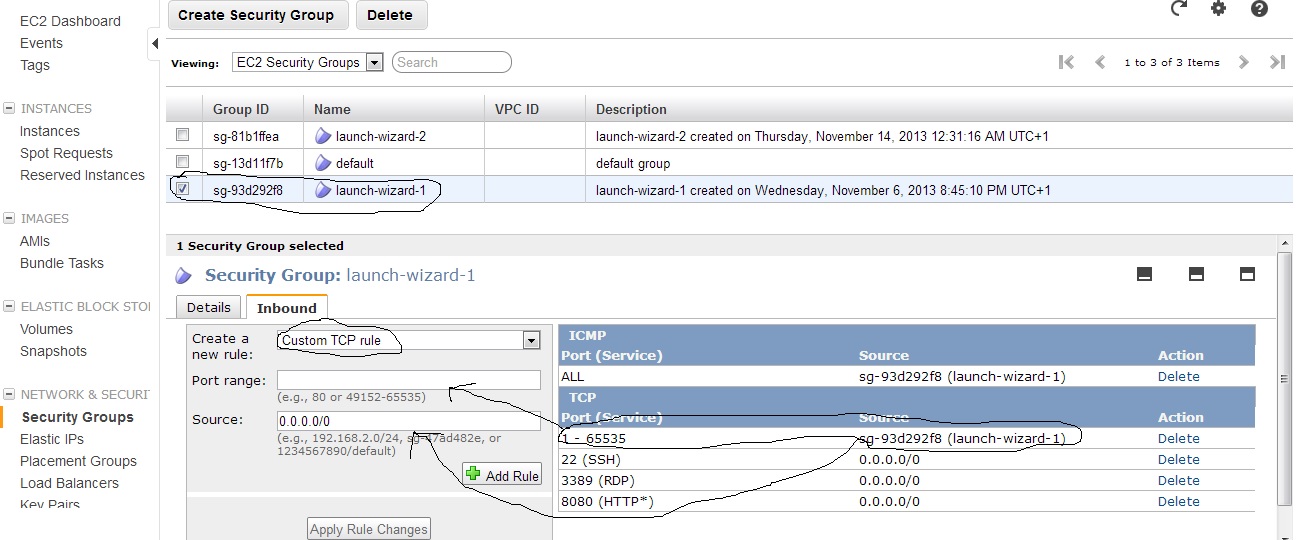
**Exericse Sheet - 10**

1. Create a new security group for custom TCP rule, allow all ports and a source as the launch wizard. This launch wizard shall be used for all three instances to be created later.



1. Create 3 EC2 instances in same availability zone. Use the security group created earlier for all three instances.
2. Install dependencies on all three instances.

sudo apt-get -y install make gcc g++ openmpi-bin openmpi-common libopenmpi-dev

1. Generate key on master

ssh-keygen -t rsa

1. Append the contend of .ssh/id\_rsa.pub (master) to .ssh/authorized\_keys(slaves)
2. Create a file hosts.mpi with the contents

master

node1

node2

1. Edit file /etc/hosts and add entry

10.45.178.13 ip-10-45-178-13 master

10.181.143.105 ip-10-181-143-105 node1

10.181.143.183 ip-10-181-143-183 node2

Where first entry is the IP address, can be found using ‘ifconfig’ command

Second entry is the host name, can be found using ‘hostname’ command

1. Create a file monte\_carlo.c with the contents at the end of the document on the master
2. Compile the program using ‘mpicc monte\_carlo.c -o monte\_carlo’ on the master
3. Open file ~/.bashrc and add the following entries at the end of the file.

export PATH=/usr/lib/openmpi:$PATH

export LD\_LIBRARY\_PATH=/usr/lib/openmpi/lib

1. Do the above change in slaves as well.
2. After the changes execute the command ‘. .bashrc’
3. Copy the compiled file to the slave nodes using the following commands

scp monte\_carlo node1:~

scp monte\_carlo node2:~

1. Now run the following command on the master

mpirun -np 6 --hostfile hosts.mpi monte\_carlo 14000

1. The last argument in the above command (14000) is the number of points used for PI estimation.
2. The above command shall print the number of points used, the PI value and the time elapsed.

**File : monte\_carlo.c**

#include <stdio.h>

#include <stdlib.h>

#include <mpi.h>

#include <math.h>

#include <time.h>

int main(int argc, char\* argv[]){

int i,id, np,N;

double x, y,double\_N,eTime,sTime,pTime;

int lhit;

MPI\_Init(&argc, &argv);

MPI\_Comm\_size(MPI\_COMM\_WORLD, &np);

MPI\_Comm\_rank(MPI\_COMM\_WORLD, &id);

if(argc != 2){

//if (id==0){

fprintf(stderr,"Wrong number of arguments \n");

fflush(stderr);

//}

MPI\_Abort(MPI\_COMM\_WORLD,1);

} else {

sscanf( argv[1], "%d", &N);

}

//sscanf(argv[1], "%lf", &double\_N);

//N = 1000;//lround(double\_N);

MPI\_Barrier(MPI\_COMM\_WORLD);

sTime = MPI\_Wtime();

lhit = 0;

srand((unsigned)(time(0)));

int lN = N/np;

for(i = 0; i<lN;i++){

x = ((double)rand())/((double)RAND\_MAX);

y = ((double)rand())/((double)RAND\_MAX);

if (((x\*x) + (y\*y)) <= 1) lhit++;

}

int hit=0;

MPI\_Allreduce(&lhit,&hit,1,MPI\_INT,MPI\_SUM,MPI\_COMM\_WORLD);

double est;

est = (hit\*4)/((double)N);

MPI\_Barrier(MPI\_COMM\_WORLD);

eTime = MPI\_Wtime();

pTime = fabs(eTime - sTime);

if (id == 0) {

printf("Number of Points Used: %d\n",N);

printf("Estimate of Pi: %24.16f\n",est);

printf("Elapsed Wall time: %5.3e\n",pTime);

}

MPI\_Finalize();

return 0;

}