PROGRAMMING ASSIGNMENT 3

CloudKon clone with Amazon SQS, DynamoDB, EC2 and S3

DESIGN:

The program is written in Java. The basic functionality is to do load balancing. Here we have to simulate the working of a client and multiple workers who will pick the job from a global queue as and when they are free. They will not be allocated work by the client as is the usual case so they wont block and can run efficiently.

In the code, a Java class called Wrapper.java is defined which is the main entry point for the code and contains if- conditional statements for the three scenarios namely Local Client simulation, Remote Client and Remote workers. The scenario is identified by the parameters passed in Command Line Arguments when the program is executed. So the same jar file is used to run client and worker for remote and local cases. It is dynamically identified.

LOCAL CLIENT/WORKER :
The jobs i.e. sleep jobs are placed in an in-memory queue implemented using LinkedList and the client itself acts as the workers and runs these jobs in a multithreaded set up. After the job is picked up and run, it its response is written to another queue. After all execution completes it writes the responses to another file called responses_local.txt.

REMOTE CLIENT:
The remote client picks up tasks from a file and writes it to the SQS queue whose name is passed by the user. After the messages are added to the queue, it polls and waits till the response queue has all the requests and when this happens, it exits.

REMOTE WORKER:
The remote worker/ workers are each multi-threaded and pick up the jobs from the SQS queue created above one at a time and perform the task mentioned. It then writes response to another SQS response queue. As we pick up the tasks from the source queue, and we are done executing the task, we delete them. This is in order to get a condition to loop over for the worker. It keeps picking the tasks from the queue, runs the jobs and deletes it. This continues as long as the source queue size does not become zero. To prevent duplication, each time we pick the task, we write the taskID to TaskTracker table in DynamoDB as a validation that we have not picked up a duplicate task. If there is a duplicate, the job is dropped and next one is picked up. (There is possibility of same job being picked up before it is deleted and this is how it is handled.)
So if more than one worker is trying to pick the same task, it will not be allowed to do so.

The timer is started when the client starts putting the messages on source queue and ends when the size of response queue equals the size of the original job queue. This is the total time taken to execute the job. Throughput and efficiency is calculated with respect to this time. So we do incur the cost of adding messages to the queue, persisting data to DynamoDB, delay in pulling the messages and writing them to response queue.

For calculating throughput, we run 10k sleep 0 jobs with one, two, four , eight and sixteen workers each of which can have multi-threading. We notice that the system scales well as number of workers increases.

Efficiency is calculated by running 10ms,1s and 10s jobs by assigning a fixed number of jobs per worker. The value is calculated using the formula: ideal time taken divided by the time taken for actually running the task.

MANUAL

Paste the work load file,.aws folder containing the credentials and the jar file given in the executables folder in the directory where the below mentioned commands will be run.

While running the 16 worker experiment, be sure to increase the provisioning limits of the dynamo DB to a higher value.

1) To run local worker give the following command:

java -jar Interface.jar client -s LOCAL -t 1 -w workload_0ms_10000.txt

Vary the parameter after -t for different number of workers
Similarly, vary the workload file name and number of threads for the other cases.

2) To run remote client and workers give the following commands:

Create the required number of t2-micro instances in Amazon EC2.

For throughput calculation:

Client:

java -jar Interface.jar client -s RemoteQ -w workload_0ms_10000.txt

Run the following command on as many EC2 instances as the required number of workers:

Worker:

java -jar Interface.jar worker -s RemoteQ -t 10

For efficiency calculations:

10ms:

Worker:
java -jar Interface.jar worker -s RemoteQ -t 10

Client for 1 worker:
java -jar Interface.jar client -s RemoteQ -w workload_10ms_1000.txt

Client for 2 workers:
java -jar Interface.jar client -s RemoteQ -w workload_10ms_2000.txt

Client for 4 workers:

java -jar Interface.jar client -s RemoteQ -w workload_10ms_4000.txt

Client for 8 workers:
java -jar Interface.jar client -s RemoteQ -w workload_10ms_8000.txt
Client for 16 workers:
java -jar Interface.jar client -s RemoteQ -w workload_10ms_16000.txt

1s:

Worker:
java -jar Interface.jar worker -s RemoteQ -t 10

Client for 1 worker:
java -jar Interface.jar client -s RemoteQ -w workload_1000ms_100.txt

Client for 2 workers:
java -jar Interface.jar client -s RemoteQ -w workload_1000ms_200.txt

Client for 4 workers:
java -jar Interface.jar client -s RemoteQ -w workload_1000ms_400.txt

Client for 8 workers:
java -jar Interface.jar client -s RemoteQ -w workload_1000ms_800.txt

Client for 16 workers:
java -jar Interface.jar client -s RemoteQ -w workload_1000ms_1600.txt

10s:

Worker:
java -jar Interface.jar worker -s RemoteQ -t 10

Client for 1 worker:
java -jar Interface.jar client -s RemoteQ -w workload_10000ms_10.txt

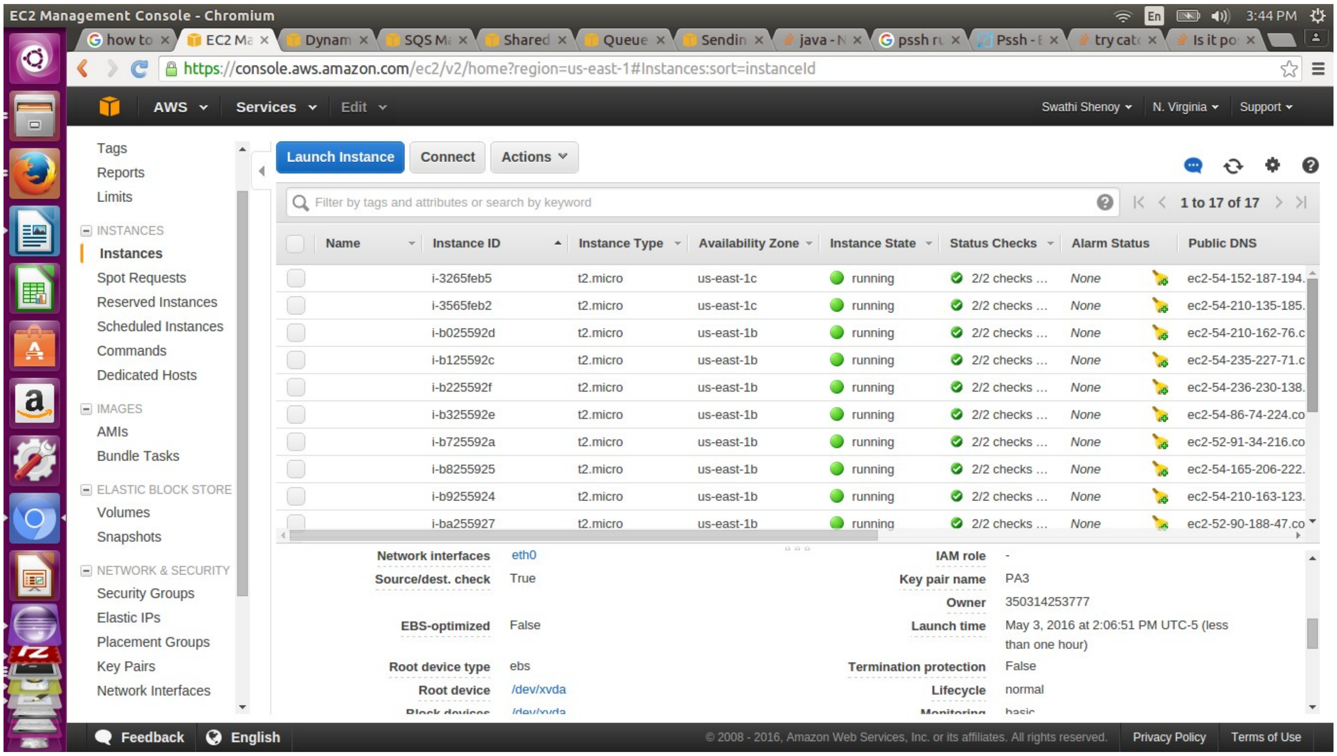
Client for 2 workers:
java -jar Interface.jar client -s RemoteQ -w workload_10000ms_20.txt

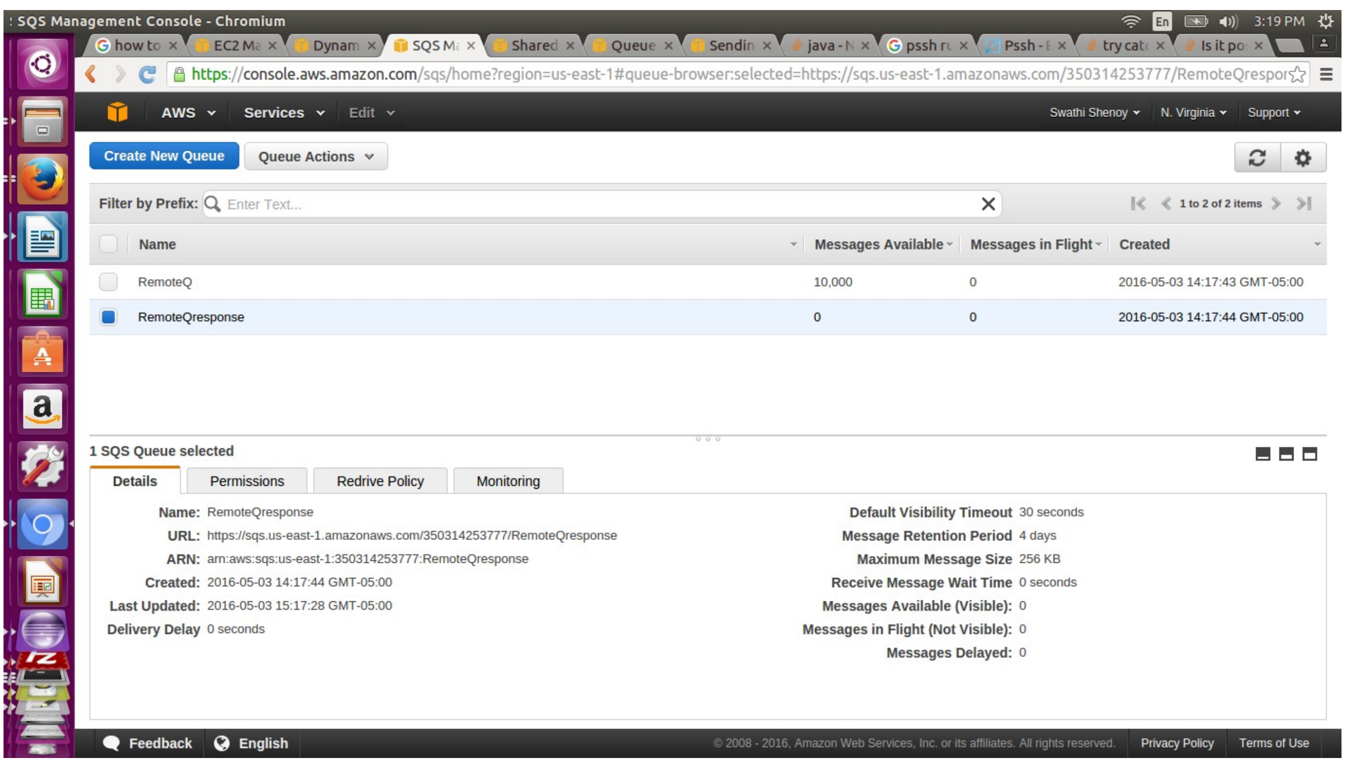
Client for 4 workers:
java -jar Interface.jar client -s RemoteQ -w workload_10000ms_40.txt

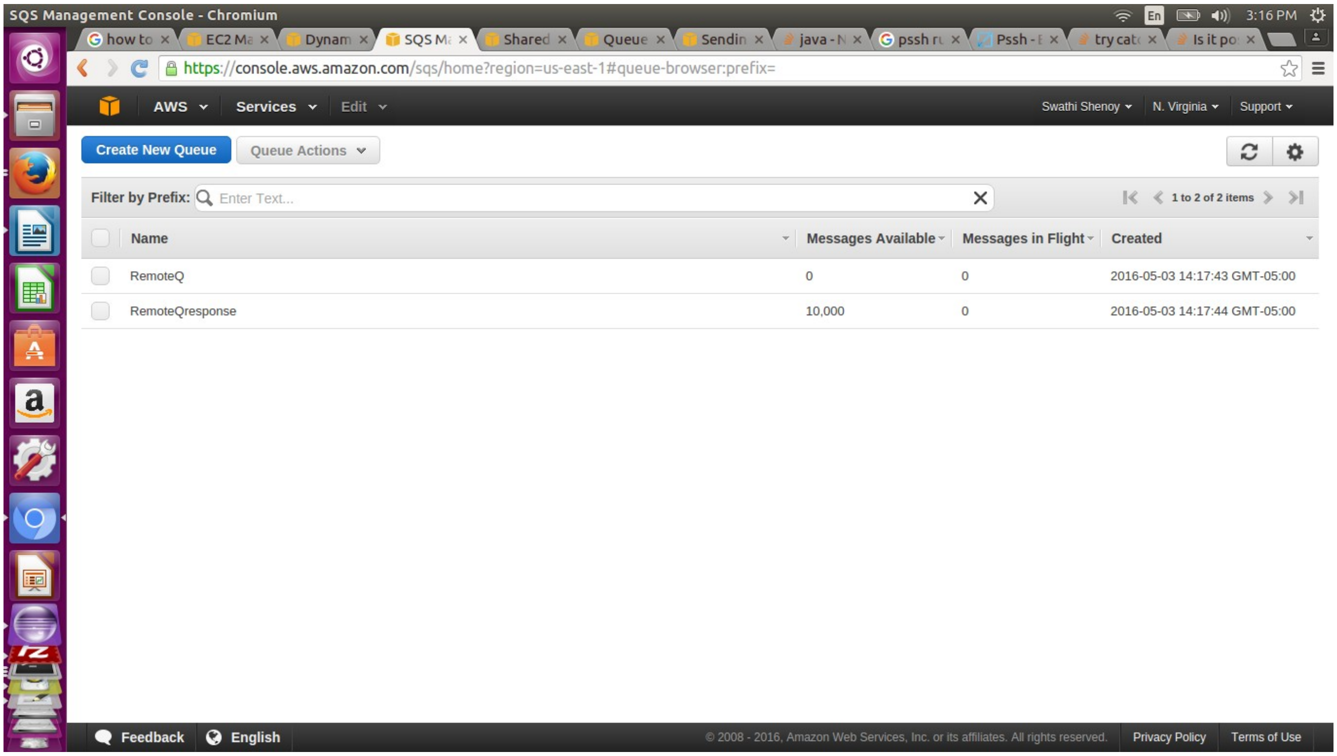
Client for 8 workers:
java -jar Interface.jar client -s RemoteQ -w workload_10000ms_80.txt

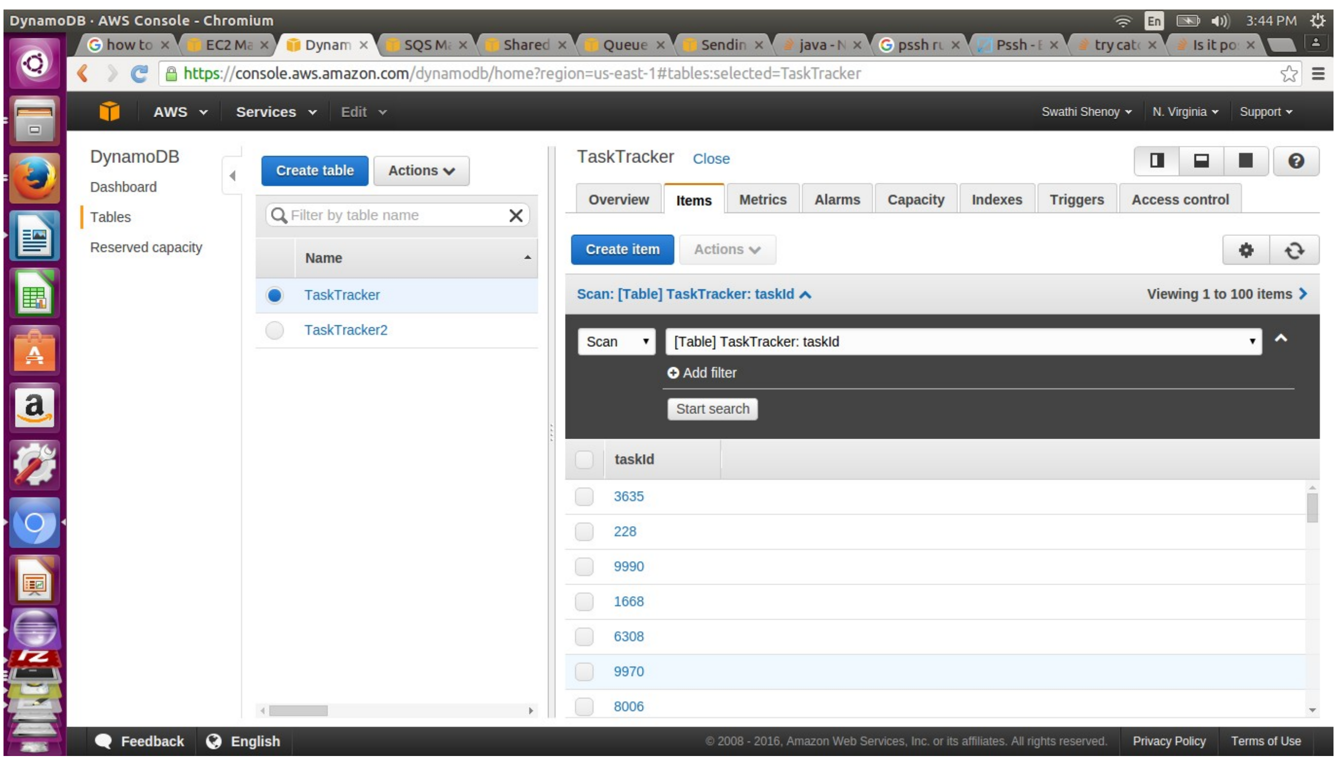
Client for 16 workers:
java -jar Interface.jar client -s RemoteQ -w workload_10000ms_160.txt

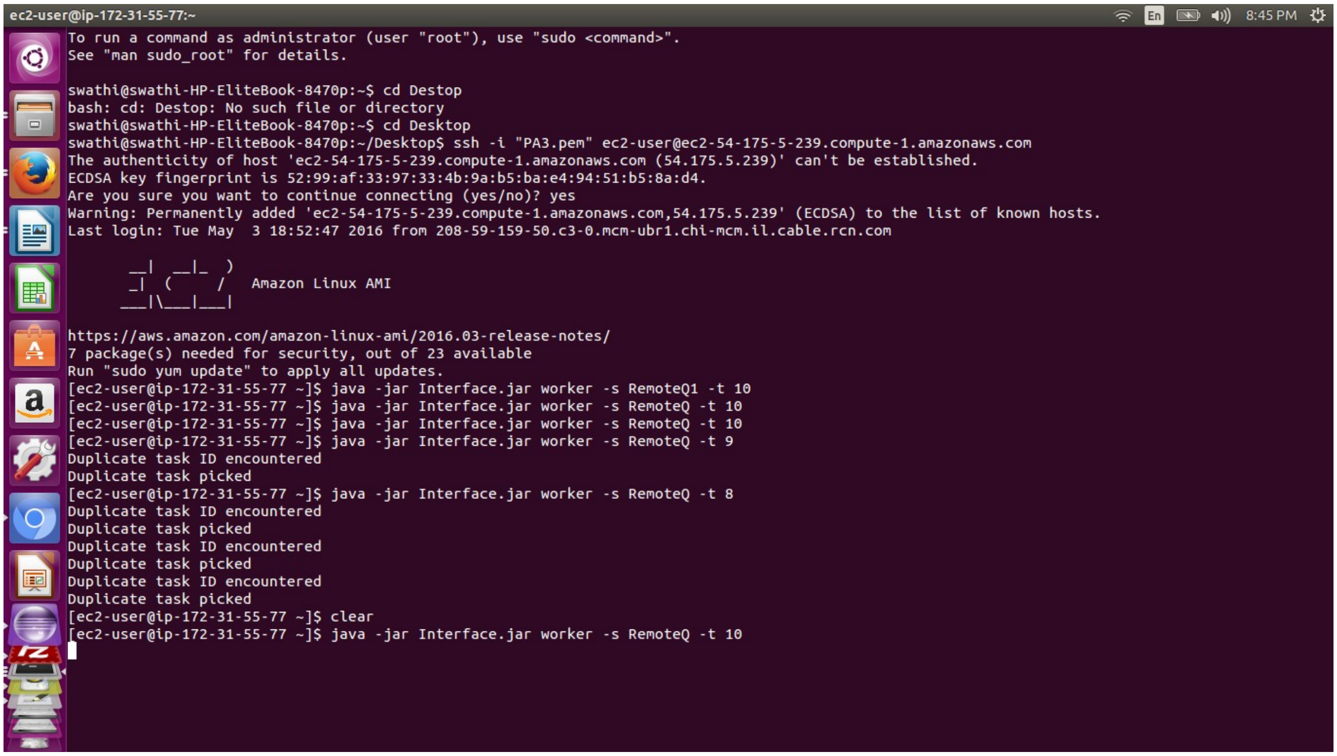
SCREENSHOTS:



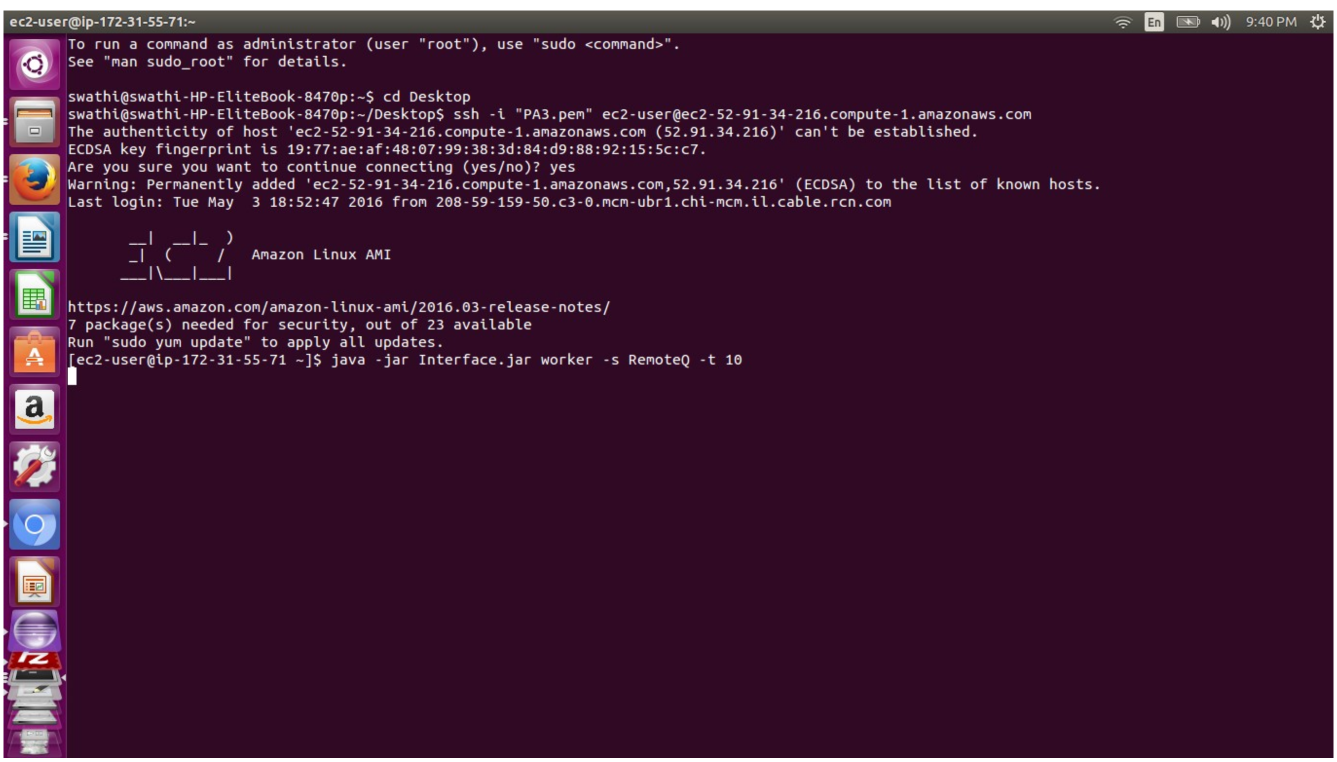



SQS Management Console - Chromium
https://console.aws.amazon.com/sqs/home?region=us-east-1#queue-browser:prefix=
AWS
Services
Edit
Swathi Shenoy
N. Virginia
Support
Create New Queue
Queue Actions
Filter by Prefix:
Enter Text...
1 to 2 of 2 items
Name
Messages Available
Messages in Flight
Created
RemoteQ 0 0 2016-05-03 14:17:43 GMT-05:00
RemoteQresponse 10,000 0 2016-05-03 14:17:44 GMT-05:00
Feedback
English
© 2008 - 2016, Amazon Web Services, Inc. or its affiliates. All rights reserved.
Privacy Policy
Terms of Use

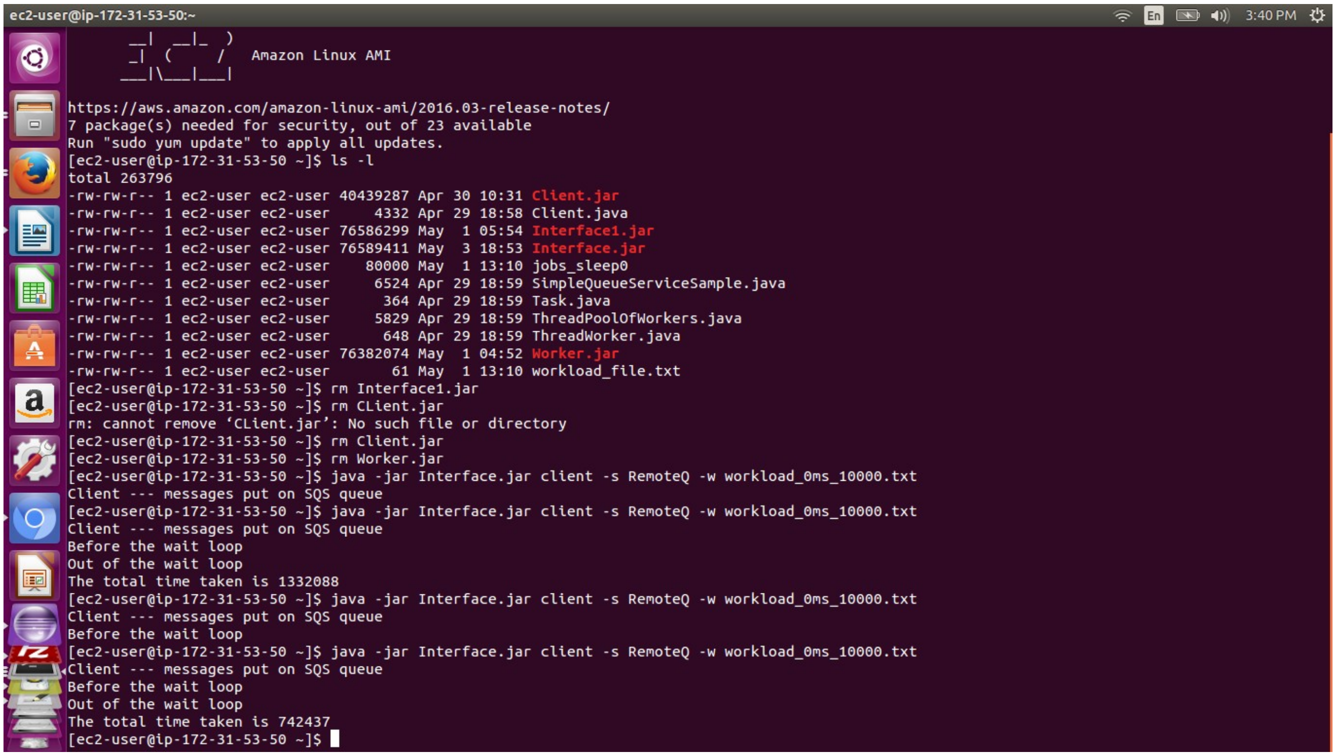

DynamoDB · AWS Console - Chromium
https://console.aws.amazon.com/dynamodb/home?region=us-east-1#tables:selected=TaskTracker
AWS
Services
Edit
Swathi Shenoy
N. Virginia
Support
DynamoDB
Dashboard
Tables
Reserved capacity
Create table
Actions
Filter by table name
Name
TaskTracker
TaskTracker2
TaskTracker
Close
Overview
Items
Metrics
Alarms
Capacity
Indexes
Triggers
Access control
Create item
Actions
Scan: [Table] TaskTracker: taskId
Viewing 1 to 100 items
Scan
[Table] TaskTracker: taskId
Add filter
Start search
taskId
3635
228
9990
1668
6308
9970
8006
Feedback
English
© 2008 - 2016, Amazon Web Services, Inc. or its affiliates. All rights reserved.
Privacy Policy
Terms of Use

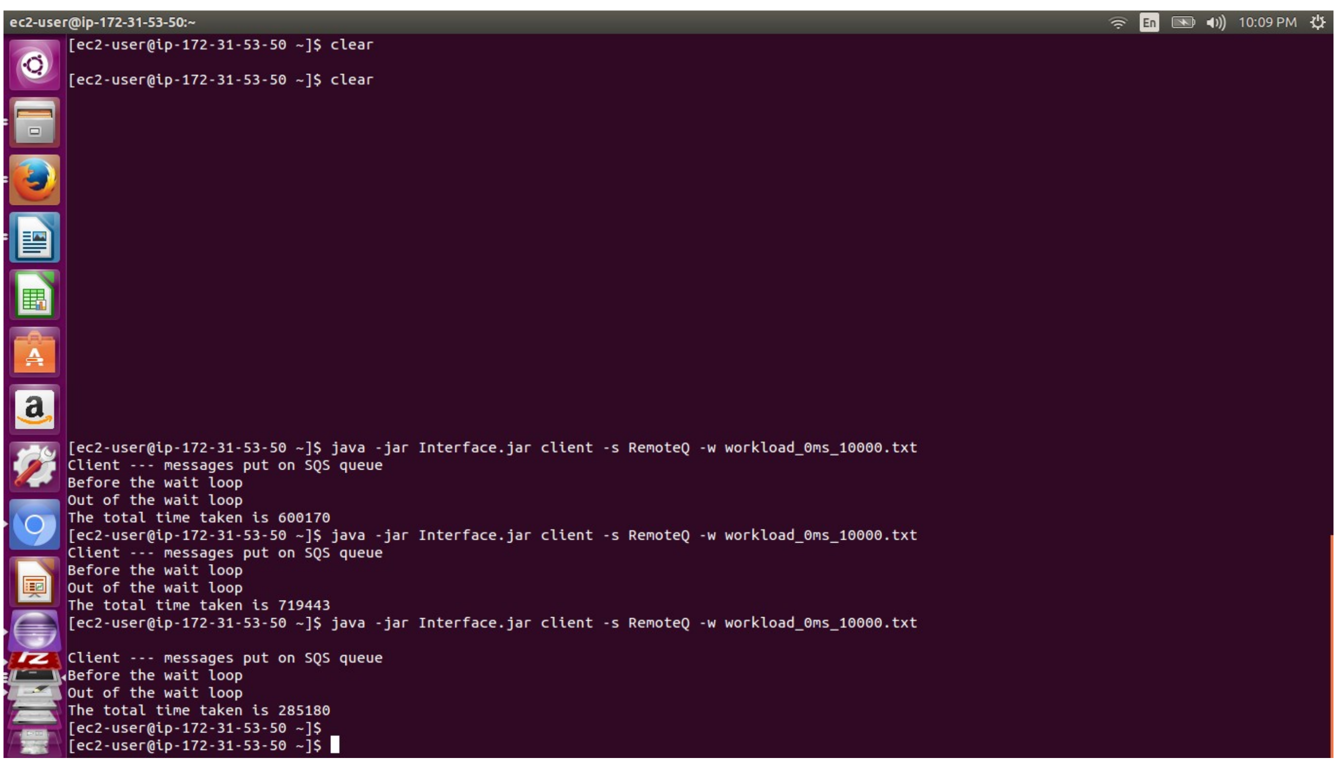


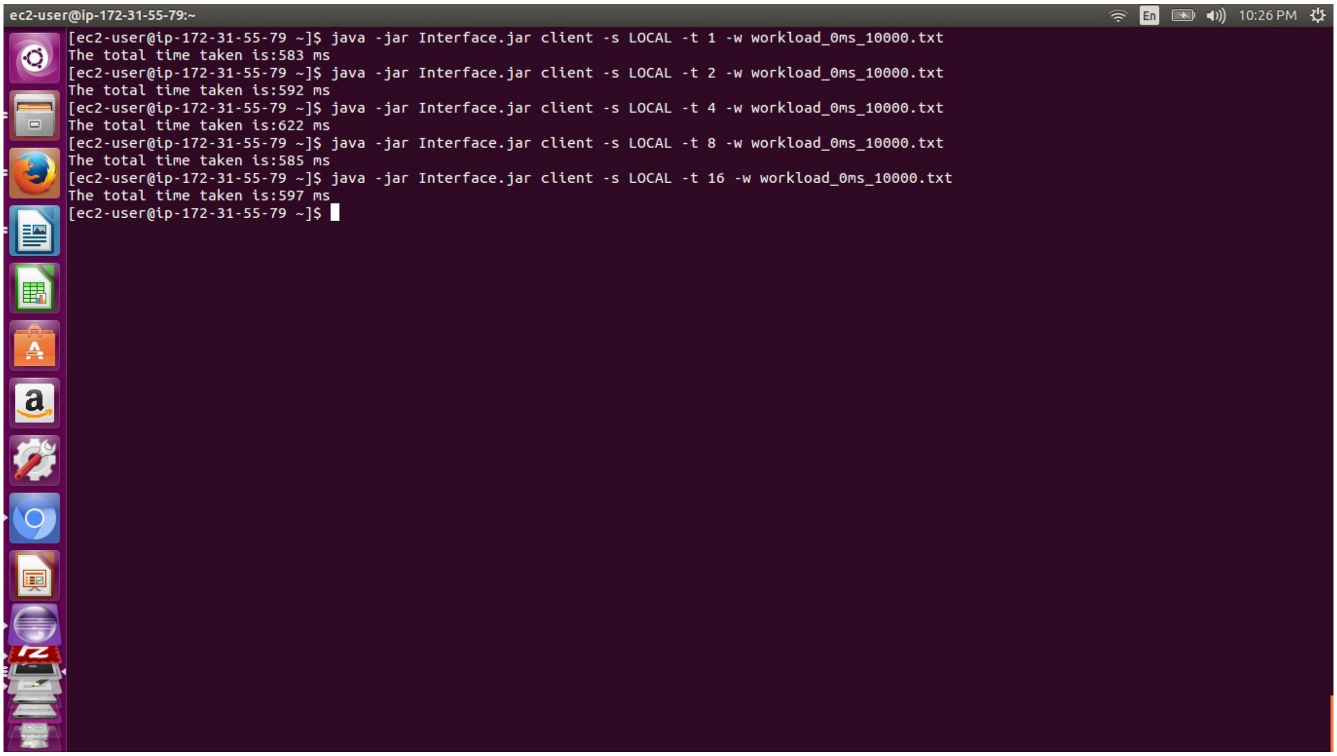

```
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

swathi@swathi-HP-EliteBook-8470p:~$ cd Destop
bash: cd: Destop: No such file or directory
swathi@swathi-HP-EliteBook-8470p:~$ cd Desktop
swathi@swathi-HP-EliteBook-8470p:~/Desktop$ ssh -i "PA3.pem" ec2-user@ec2-54-175-5-239.compute-1.amazonaws.com
The authenticity of host 'ec2-54-175-5-239.compute-1.amazonaws.com (54.175.5.239)' can't be established.
ECDSA key fingerprint is 52:99:af:33:97:33:4b:9a:b5:ba:e4:94:51:b5:8a:d4.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'ec2-54-175-5-239.compute-1.amazonaws.com,54.175.5.239' (ECDSA) to the list of known hosts.
Last login: Tue May  3 18:52:47 2016 from 208-59-159-50.c3-0.mcm-ubr1.chi-mcm.il.cable.rcn.com

       __|  __|_  )
       _|  (     /   Amazon Linux AMI
      ___|\___|___|

https://aws.amazon.com/amazon-linux-ami/2016.03-release-notes/
7 package(s) needed for security, out of 23 available
Run "sudo yum update" to apply all updates.
[ec2-user@ip-172-31-55-77 ~]$ java -jar Interface.jar worker -s RemoteQ1 -t 10
[ec2-user@ip-172-31-55-77 ~]$ java -jar Interface.jar worker -s RemoteQ -t 10
[ec2-user@ip-172-31-55-77 ~]$ java -jar Interface.jar worker -s RemoteQ -t 10
[ec2-user@ip-172-31-55-77 ~]$ java -jar Interface.jar worker -s RemoteQ -t 9
Duplicate task ID encountered
Duplicate task picked
[ec2-user@ip-172-31-55-77 ~]$ java -jar Interface.jar worker -s RemoteQ -t 8
Duplicate task ID encountered
Duplicate task picked
Duplicate task ID encountered
Duplicate task picked
Duplicate task ID encountered
Duplicate task picked
[ec2-user@ip-172-31-55-77 ~]$ clear
[ec2-user@ip-172-31-55-77 ~]$ java -jar Interface.jar worker -s RemoteQ -t 10
```

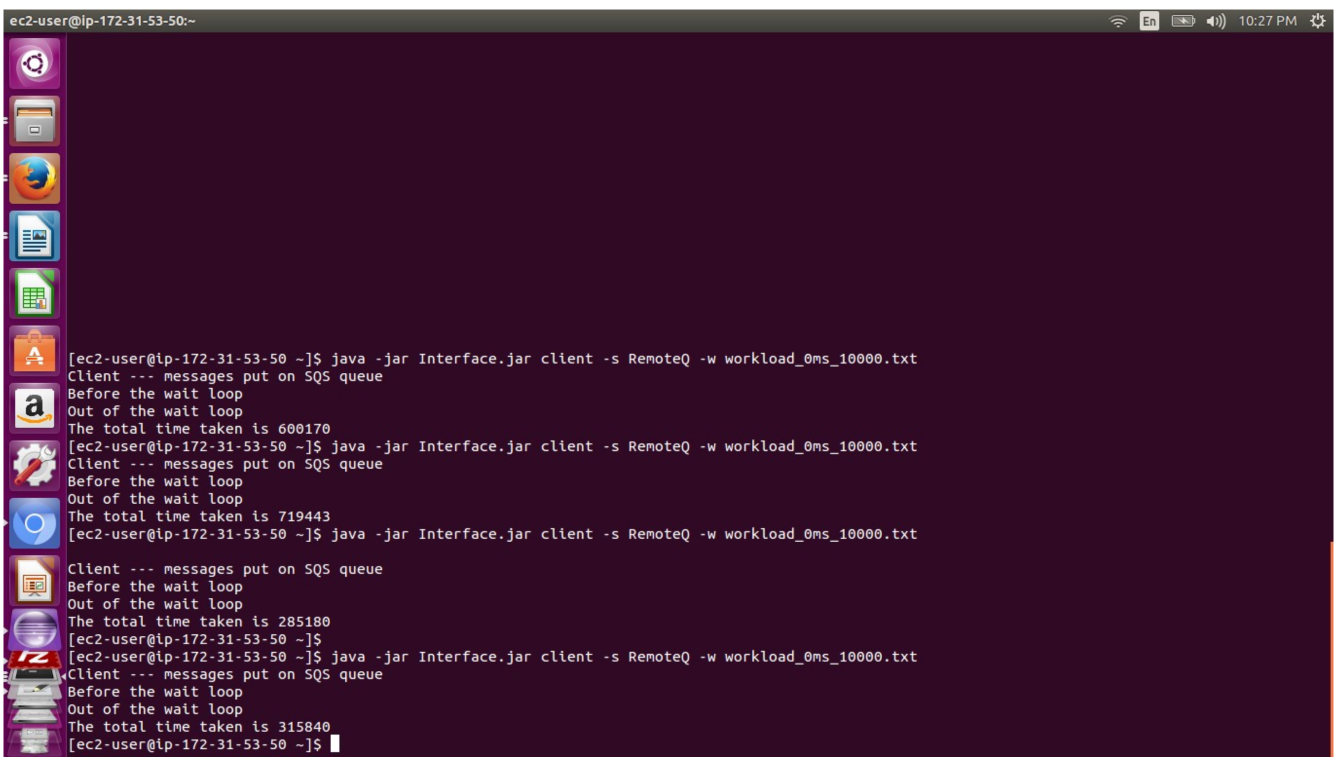


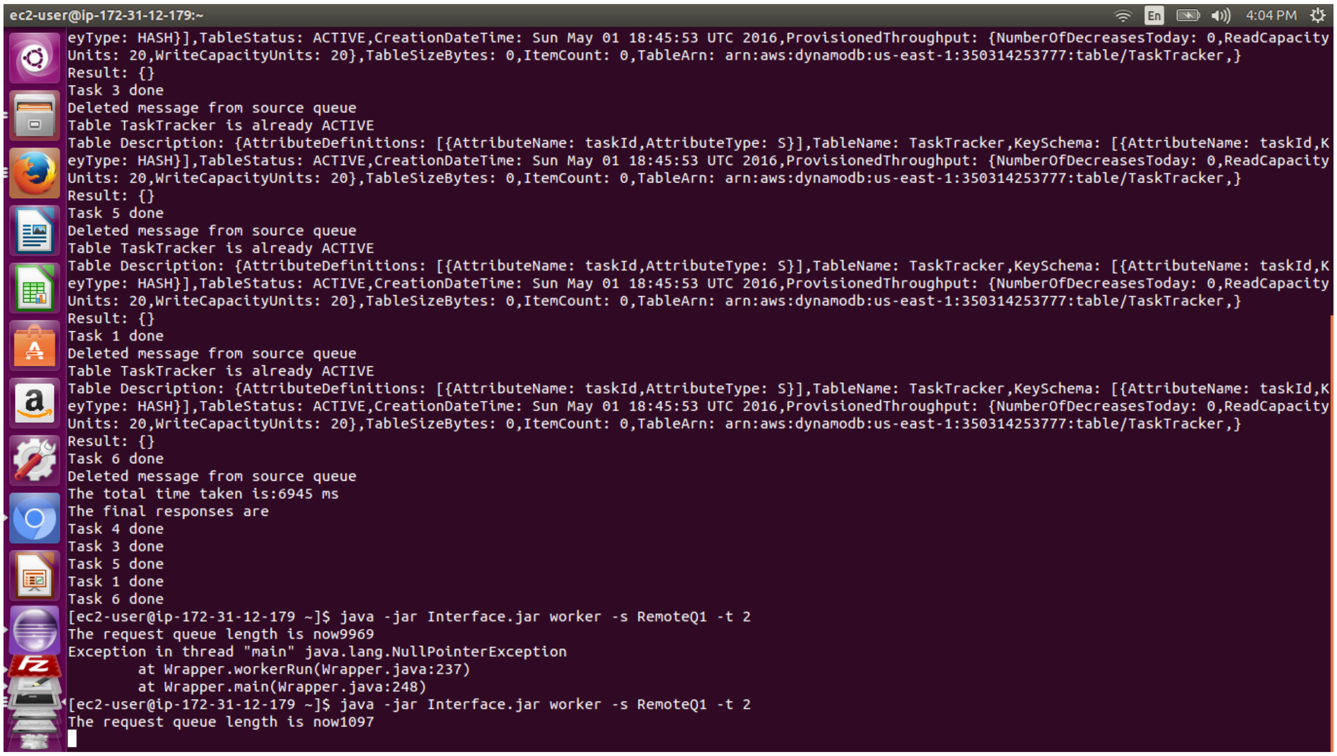

```
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

swathi@swathi-HP-EliteBook-8470p:~$ cd Desktop
swathi@swathi-HP-EliteBook-8470p:~/Desktop$ ssh -i "PA3.pem" ec2-user@ec2-52-91-34-216.compute-1.amazonaws.com
The authenticity of host 'ec2-52-91-34-216.compute-1.amazonaws.com (52.91.34.216)' can't be established.
ECDSA key fingerprint is 19:77:ae:af:48:07:99:38:3d:84:d9:88:92:15:5c:c7.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'ec2-52-91-34-216.compute-1.amazonaws.com,52.91.34.216' (ECDSA) to the list of known hosts.
Last login: Tue May  3 18:52:47 2016 from 208-59-159-50.c3-0.mcm-ubr1.chi-mcm.il.cable.rcn.com

       __|  __|_  )
       _|  (     /   Amazon Linux AMI
      ___|\___|___|

https://aws.amazon.com/amazon-linux-ami/2016.03-release-notes/
7 package(s) needed for security, out of 23 available
Run "sudo yum update" to apply all updates.
[ec2-user@ip-172-31-55-71 ~]$ java -jar Interface.jar worker -s RemoteQ -t 10
```

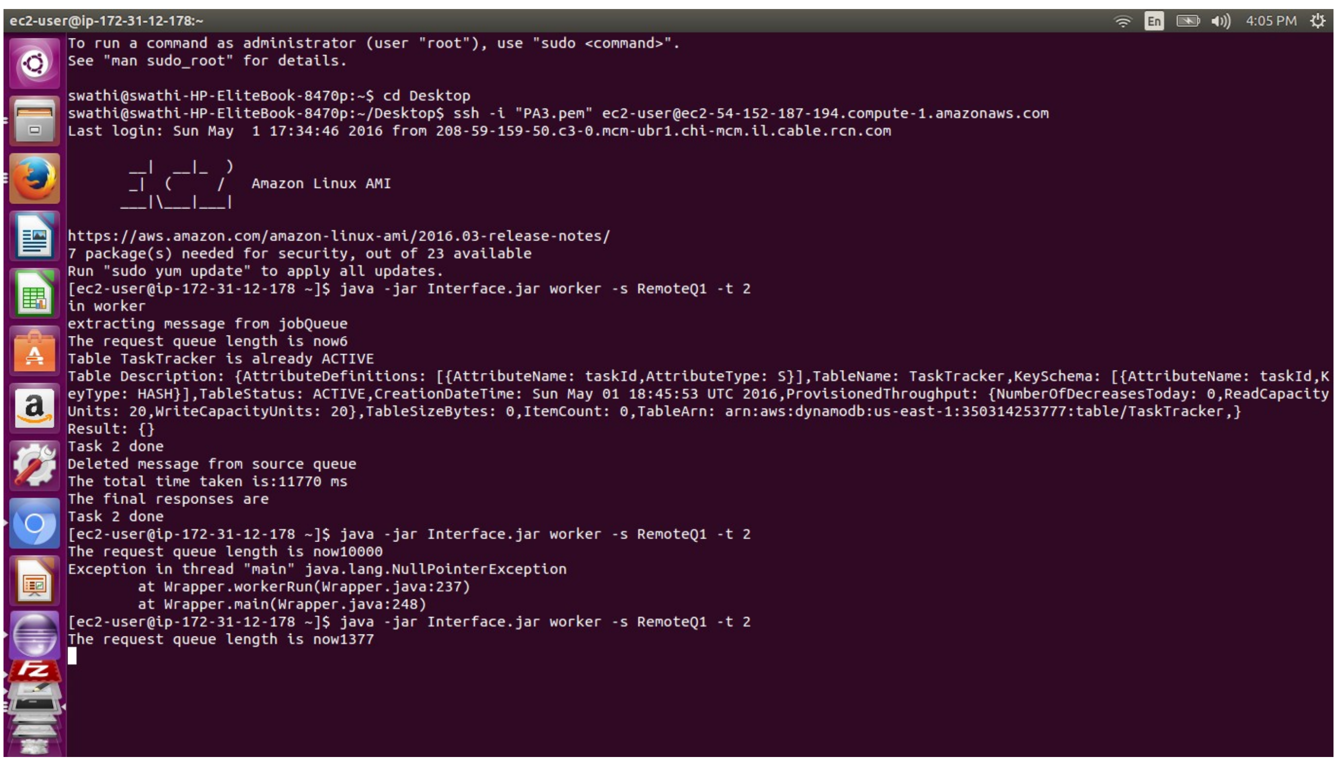

ec2-user@ip-172-31-53-50:~
3:40 PM
Amazon Linux AMI
https://aws.amazon.com/amazon-linux-ami/2016.03-release-notes/
7 package(s) needed for security, out of 23 available
Run "sudo yum update" to apply all updates.
[ec2-user@ip-172-31-53-50 ~]$ ls -l
total 263796
-rw-rw-r-- 1 ec2-user ec2-user 40439287 Apr 30 10:31 Client.jar
-rw-rw-r-- 1 ec2-user ec2-user 4332 Apr 29 18:58 Client.java
-rw-rw-r-- 1 ec2-user ec2-user 76586299 May 1 05:54 Interface1.jar
-rw-rw-r-- 1 ec2-user ec2-user 76589411 May 3 18:53 Interface.jar
-rw-rw-r-- 1 ec2-user ec2-user 80000 May 1 13:10 jobs_sleep0
-rw-rw-r-- 1 ec2-user ec2-user 6524 Apr 29 18:59 SimpleQueueServiceSample.java
-rw-rw-r-- 1 ec2-user ec2-user 364 Apr 29 18:59 Task.java
-rw-rw-r-- 1 ec2-user ec2-user 5829 Apr 29 18:59 ThreadPoolOfWorkers.java
-rw-rw-r-- 1 ec2-user ec2-user 648 Apr 29 18:59 ThreadWorker.java
-rw-rw-r-- 1 ec2-user ec2-user 76382074 May 1 04:52 Worker.jar
-rw-rw-r-- 1 ec2-user ec2-user 61 May 1 13:10 workload_file.txt
[ec2-user@ip-172-31-53-50 ~]$ rm Interface1.jar
[ec2-user@ip-172-31-53-50 ~]$ rm CLient.jar
rm: cannot remove ‘CLient.jar’: No such file or directory
[ec2-user@ip-172-31-53-50 ~]$ rm Client.jar
[ec2-user@ip-172-31-53-50 ~]$ rm Worker.jar
[ec2-user@ip-172-31-53-50 ~]$ java -jar Interface.jar client -s RemoteQ -w workload_0ms_10000.txt
Client --- messages put on SQS queue
[ec2-user@ip-172-31-53-50 ~]$ java -jar Interface.jar client -s RemoteQ -w workload_0ms_10000.txt
Client --- messages put on SQS queue
Before the wait loop
Out of the wait loop
The total time taken is 1332088
[ec2-user@ip-172-31-53-50 ~]$ java -jar Interface.jar client -s RemoteQ -w workload_0ms_10000.txt
Client --- messages put on SQS queue
Before the wait loop
[ec2-user@ip-172-31-53-50 ~]$ java -jar Interface.jar client -s RemoteQ -w workload_0ms_10000.txt
Client --- messages put on SQS queue
Before the wait loop
Out of the wait loop
The total time taken is 742437
[ec2-user@ip-172-31-53-50 ~]$

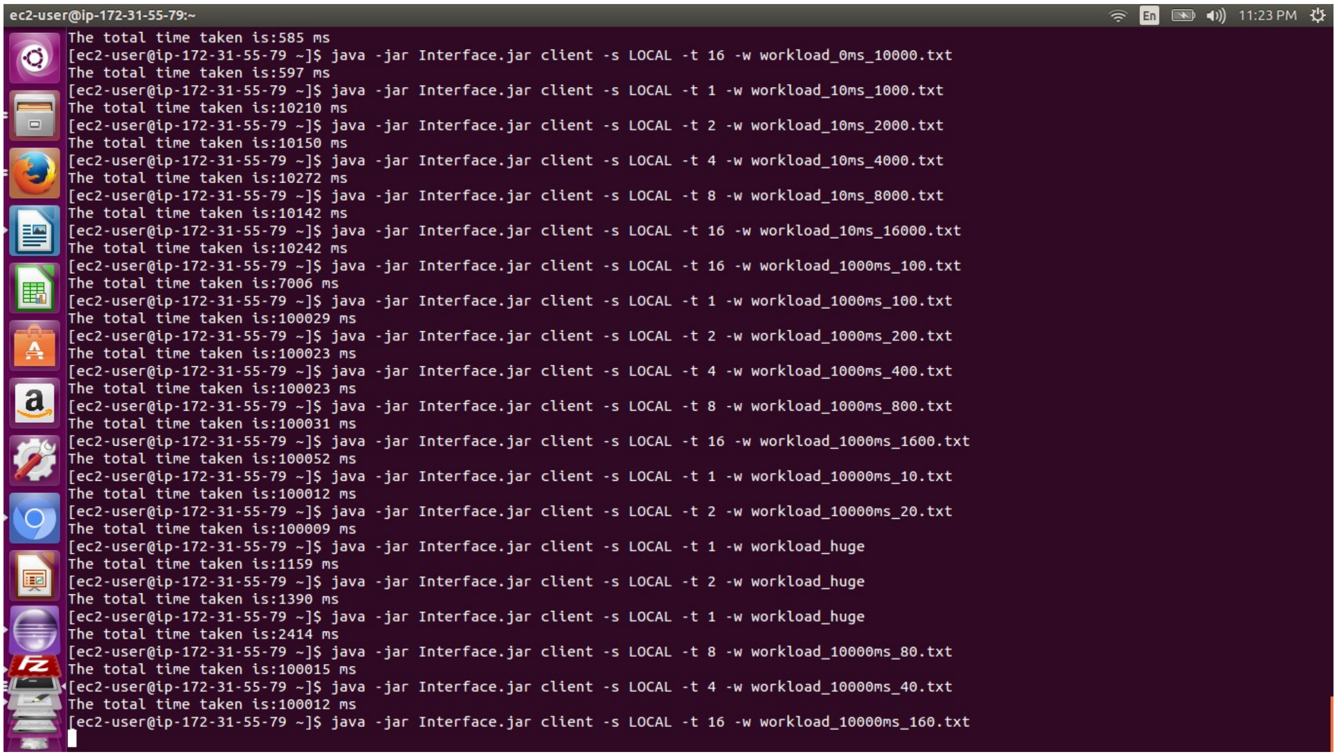

ec2-user@ip-172-31-53-50:~
10:09 PM
[ec2-user@ip-172-31-53-50 ~]$ clear
[ec2-user@ip-172-31-53-50 ~]$ clear
[ec2-user@ip-172-31-53-50 ~]$ java -jar Interface.jar client -s RemoteQ -w workload_0ms_10000.txt
Client --- messages put on SQS queue
Before the wait loop
Out of the wait loop
The total time taken is 600170
[ec2-user@ip-172-31-53-50 ~]$ java -jar Interface.jar client -s RemoteQ -w workload_0ms_10000.txt
Client --- messages put on SQS queue
Before the wait loop
Out of the wait loop
The total time taken is 719443
[ec2-user@ip-172-31-53-50 ~]$ java -jar Interface.jar client -s RemoteQ -w workload_0ms_10000.txt
Client --- messages put on SQS queue
Before the wait loop
Out of the wait loop
The total time taken is 285180
[ec2-user@ip-172-31-53-50 ~]$
[ec2-user@ip-172-31-53-50 ~]$

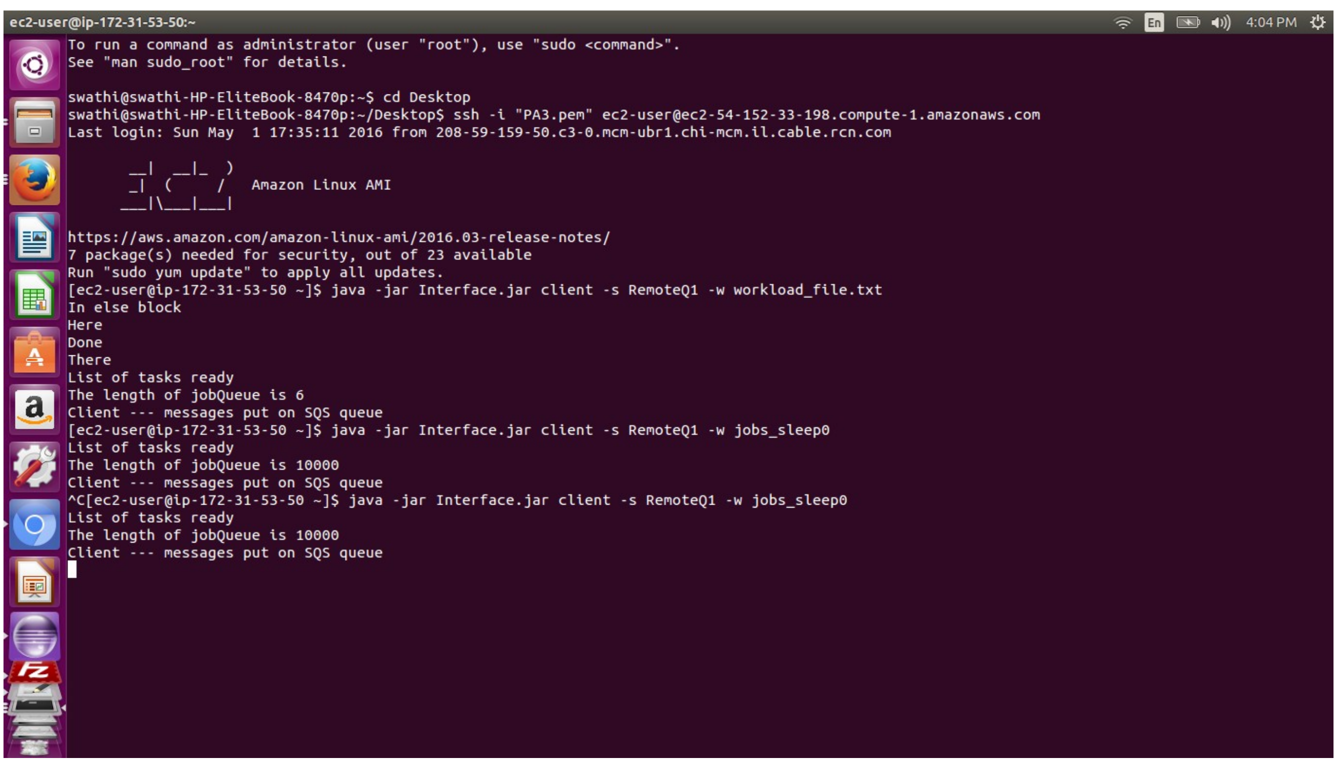

ec2-user@ip-172-31-55-79:~
10:26 PM
[ec2-user@ip-172-31-55-79 ~]$ java -jar Interface.jar client -s LOCAL -t 1 -w workload_0ms_10000.txt
The total time taken is:583 ms
[ec2-user@ip-172-31-55-79 ~]$ java -jar Interface.jar client -s LOCAL -t 2 -w workload_0ms_10000.txt
The total time taken is:592 ms
[ec2-user@ip-172-31-55-79 ~]$ java -jar Interface.jar client -s LOCAL -t 4 -w workload_0ms_10000.txt
The total time taken is:622 ms
[ec2-user@ip-172-31-55-79 ~]$ java -jar Interface.jar client -s LOCAL -t 8 -w workload_0ms_10000.txt
The total time taken is:585 ms
[ec2-user@ip-172-31-55-79 ~]$ java -jar Interface.jar client -s LOCAL -t 16 -w workload_0ms_10000.txt
The total time taken is:597 ms
[ec2-user@ip-172-31-55-79 ~]$

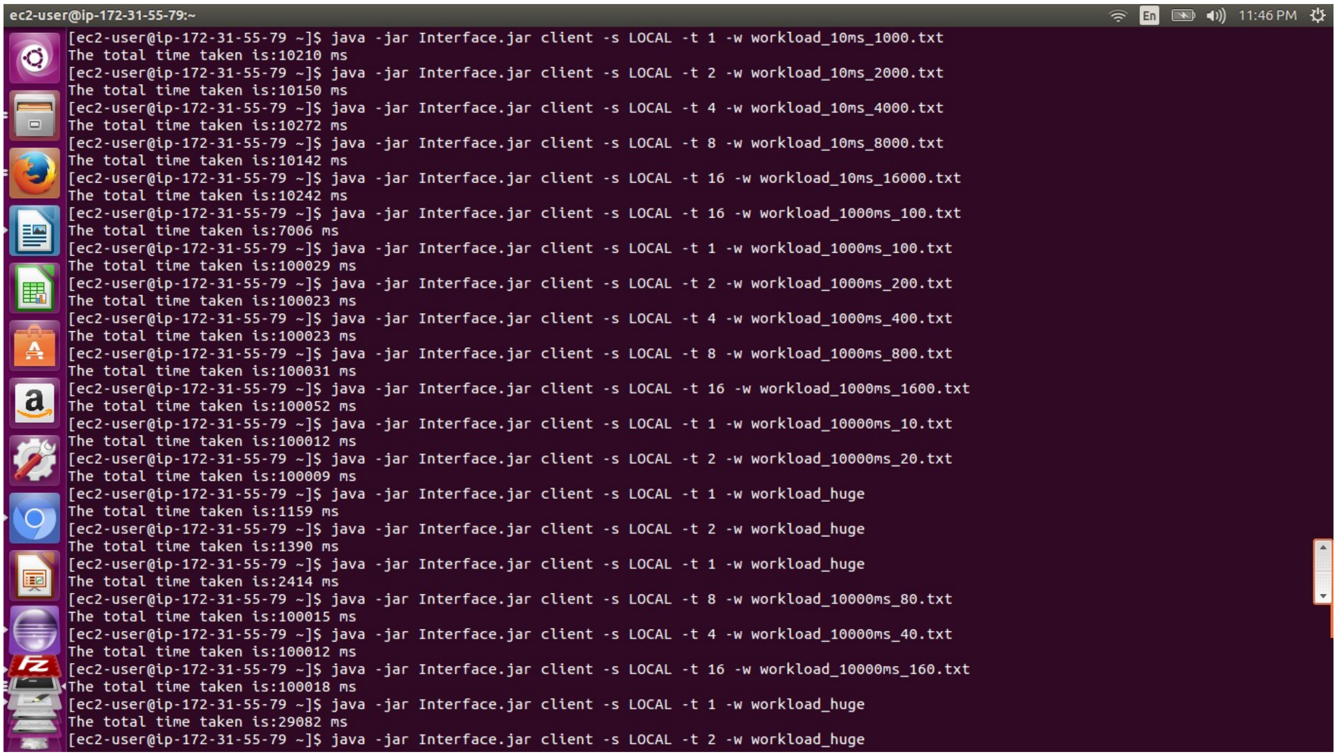

ec2-user@ip-172-31-53-50:~
10:27 PM
[ec2-user@ip-172-31-53-50 ~]$ java -jar Interface.jar client -s RemoteQ -w workload_0ms_10000.txt
Client --- messages put on SQS queue
Before the wait loop
Out of the wait loop
The total time taken is 600170
[ec2-user@ip-172-31-53-50 ~]$ java -jar Interface.jar client -s RemoteQ -w workload_0ms_10000.txt
Client --- messages put on SQS queue
Before the wait loop
Out of the wait loop
The total time taken is 719443
[ec2-user@ip-172-31-53-50 ~]$ java -jar Interface.jar client -s RemoteQ -w workload_0ms_10000.txt
Client --- messages put on SQS queue
Before the wait loop
Out of the wait loop
The total time taken is 285180
[ec2-user@ip-172-31-53-50 ~]$
[ec2-user@ip-172-31-53-50 ~]$ java -jar Interface.jar client -s RemoteQ -w workload_0ms_10000.txt
Client --- messages put on SQS queue
Before the wait loop
Out of the wait loop
The total time taken is 315840
[ec2-user@ip-172-31-53-50 ~]$

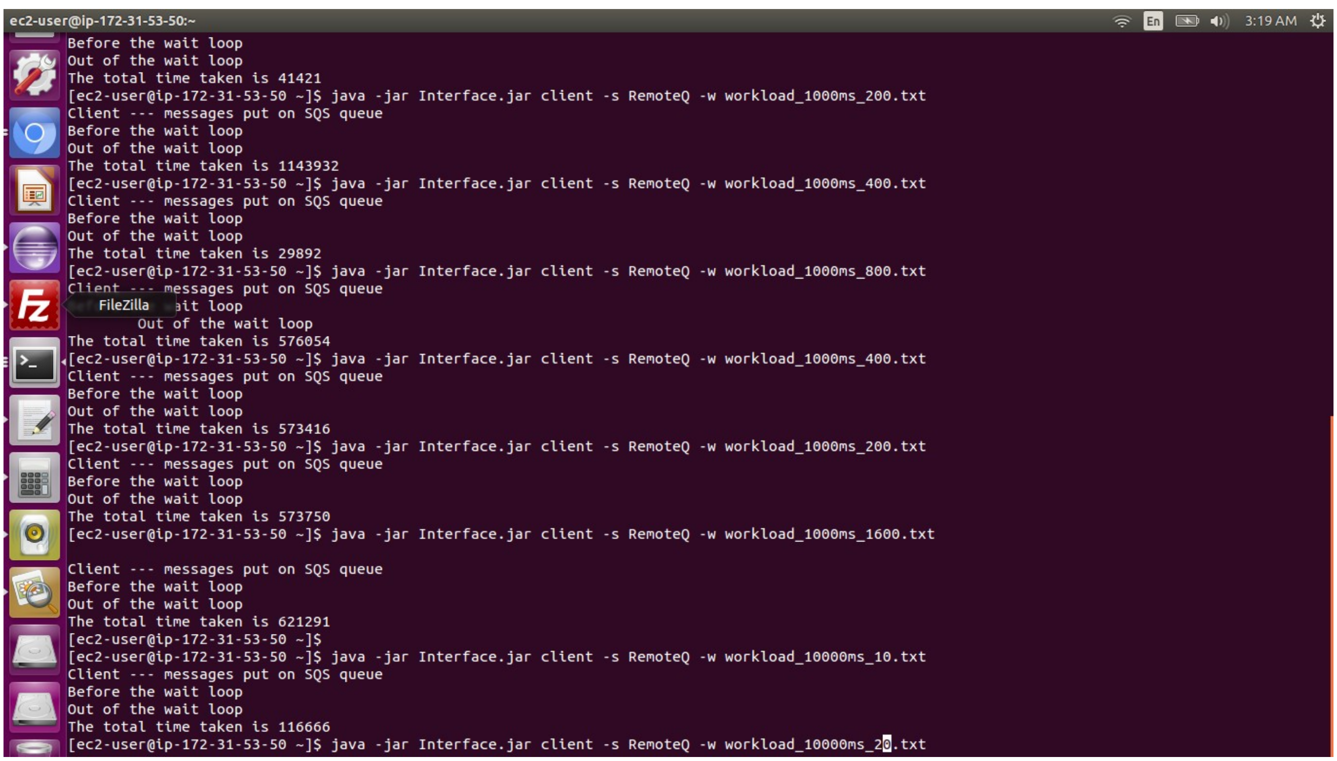

ec2-user@ip-172-31-12-179:~
4:04 PM
eyType: HASH}],TableStatus: ACTIVE,CreationDateTime: Sun May 01 18:45:53 UTC 2016,ProvisionedThroughput: {NumberOfDecreasesToday: 0,ReadCapacity
Units: 20,WriteCapacityUnits: 20},TableSizeBytes: 0,ItemCount: 0,TableArn: arn:aws:dynamodb:us-east-1:350314253777:table/TaskTracker,}
Result: {}
Task 3 done
Deleted message from source queue
Table TaskTracker is already ACTIVE
Table Description: {AttributeDefinitions: [{AttributeName: taskId,AttributeType: S}],TableName: TaskTracker,KeySchema: [{AttributeName: taskId,K
eyType: HASH}],TableStatus: ACTIVE,CreationDateTime: Sun May 01 18:45:53 UTC 2016,ProvisionedThroughput: {NumberOfDecreasesToday: 0,ReadCapacity
Units: 20,WriteCapacityUnits: 20},TableSizeBytes: 0,ItemCount: 0,TableArn: arn:aws:dynamodb:us-east-1:350314253777:table/TaskTracker,}
Result: {}
Task 5 done
Deleted message from source queue
Table TaskTracker is already ACTIVE
Table Description: {AttributeDefinitions: [{AttributeName: taskId,AttributeType: S}],TableName: TaskTracker,KeySchema: [{AttributeName: taskId,K
eyType: HASH}],TableStatus: ACTIVE,CreationDateTime: Sun May 01 18:45:53 UTC 2016,ProvisionedThroughput: {NumberOfDecreasesToday: 0,ReadCapacity
Units: 20,WriteCapacityUnits: 20},TableSizeBytes: 0,ItemCount: 0,TableArn: arn:aws:dynamodb:us-east-1:350314253777:table/TaskTracker,}
Result: {}
Task 1 done
Deleted message from source queue
Table TaskTracker is already ACTIVE
Table Description: {AttributeDefinitions: [{AttributeName: taskId,AttributeType: S}],TableName: TaskTracker,KeySchema: [{AttributeName: taskId,K
eyType: HASH}],TableStatus: ACTIVE,CreationDateTime: Sun May 01 18:45:53 UTC 2016,ProvisionedThroughput: {NumberOfDecreasesToday: 0,ReadCapacity
Units: 20,WriteCapacityUnits: 20},TableSizeBytes: 0,ItemCount: 0,TableArn: arn:aws:dynamodb:us-east-1:350314253777:table/TaskTracker,}
Result: {}
Task 6 done
Deleted message from source queue
The total time taken is:6945 ms
The final responses are
Task 4 done
Task 3 done
Task 5 done
Task 1 done
Task 6 done
[ec2-user@ip-172-31-12-179 ~]$ java -jar Interface.jar worker -s RemoteQ1 -t 2
The request queue length is now9969
Exception in thread "main" java.lang.NullPointerException
at Wrapper.workerRun(Wrapper.java:237)
at Wrapper.main(Wrapper.java:248)
[ec2-user@ip-172-31-12-179 ~]$ java -jar Interface.jar worker -s RemoteQ1 -t 2
The request queue length is now1097


ec2-user@ip-172-31-12-178:~
4:05 PM
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.
swathi@swathi-HP-EliteBook-8470p:~$ cd Desktop
swathi@swathi-HP-EliteBook-8470p:~/Desktop$ ssh -i "PA3.pem" ec2-user@ec2-54-152-187-194.compute-1.amazonaws.com
Last login: Sun May 1 17:34:46 2016 from 208-59-159-50.c3-0.mcm-ubr1.chi-mcm.il.cable.rcn.com
Amazon Linux AMI
https://aws.amazon.com/amazon-linux-ami/2016.03-release-notes/
7 package(s) needed for security, out of 23 available
Run "sudo yum update" to apply all updates.
[ec2-user@ip-172-31-12-178 ~]$ java -jar Interface.jar worker -s RemoteQ1 -t 2
in worker
extracting message from jobQueue
The request queue length is now6
Table TaskTracker is already ACTIVE
Table Description: {AttributeDefinitions: [{AttributeName: taskId,AttributeType: S}],TableName: TaskTracker,KeySchema: [{AttributeName: taskId,K
eyType: HASH}],TableStatus: ACTIVE,CreationDateTime: Sun May 01 18:45:53 UTC 2016,ProvisionedThroughput: {NumberOfDecreasesToday: 0,ReadCapacity
Units: 20,WriteCapacityUnits: 20},TableSizeBytes: 0,ItemCount: 0,TableArn: arn:aws:dynamodb:us-east-1:350314253777:table/TaskTracker,}
Result: {}
Task 2 done
Deleted message from source queue
The total time taken is:11770 ms
The final responses are
Task 2 done
[ec2-user@ip-172-31-12-178 ~]$ java -jar Interface.jar worker -s RemoteQ1 -t 2
The request queue length is now10000
Exception in thread "main" java.lang.NullPointerException
at Wrapper.workerRun(Wrapper.java:237)
at Wrapper.main(Wrapper.java:248)
[ec2-user@ip-172-31-12-178 ~]$ java -jar Interface.jar worker -s RemoteQ1 -t 2
The request queue length is now1377


ec2-user@ip-172-31-55-79:~
11:23 PM
The total time taken is:585 ms
[ec2-user@ip-172-31-55-79 ~]$ java -jar Interface.jar client -s LOCAL -t 16 -w workload_0ms_10000.txt
The total time taken is:597 ms
[ec2-user@ip-172-31-55-79 ~]$ java -jar Interface.jar client -s LOCAL -t 1 -w workload_10ms_1000.txt
The total time taken is:10210 ms
[ec2-user@ip-172-31-55-79 ~]$ java -jar Interface.jar client -s LOCAL -t 2 -w workload_10ms_2000.txt
The total time taken is:10150 ms
[ec2-user@ip-172-31-55-79 ~]$ java -jar Interface.jar client -s LOCAL -t 4 -w workload_10ms_4000.txt
The total time taken is:10272 ms
[ec2-user@ip-172-31-55-79 ~]$ java -jar Interface.jar client -s LOCAL -t 8 -w workload_10ms_8000.txt
The total time taken is:10142 ms
[ec2-user@ip-172-31-55-79 ~]$ java -jar Interface.jar client -s LOCAL -t 16 -w workload_10ms_16000.txt
The total time taken is:10242 ms
[ec2-user@ip-172-31-55-79 ~]$ java -jar Interface.jar client -s LOCAL -t 16 -w workload_1000ms_100.txt
The total time taken is:7006 ms
[ec2-user@ip-172-31-55-79 ~]$ java -jar Interface.jar client -s LOCAL -t 1 -w workload_1000ms_100.txt
The total time taken is:100029 ms
[ec2-user@ip-172-31-55-79 ~]$ java -jar Interface.jar client -s LOCAL -t 2 -w workload_1000ms_200.txt
The total time taken is:100023 ms
[ec2-user@ip-172-31-55-79 ~]$ java -jar Interface.jar client -s LOCAL -t 4 -w workload_1000ms_400.txt
The total time taken is:100023 ms
[ec2-user@ip-172-31-55-79 ~]$ java -jar Interface.jar client -s LOCAL -t 8 -w workload_1000ms_800.txt
The total time taken is:100031 ms
[ec2-user@ip-172-31-55-79 ~]$ java -jar Interface.jar client -s LOCAL -t 16 -w workload_1000ms_1600.txt
The total time taken is:100052 ms
[ec2-user@ip-172-31-55-79 ~]$ java -jar Interface.jar client -s LOCAL -t 1 -w workload_10000ms_10.txt
The total time taken is:100012 ms
[ec2-user@ip-172-31-55-79 ~]$ java -jar Interface.jar client -s LOCAL -t 2 -w workload_10000ms_20.txt
The total time taken is:100009 ms
[ec2-user@ip-172-31-55-79 ~]$ java -jar Interface.jar client -s LOCAL -t 1 -w workload_huge
The total time taken is:1159 ms
[ec2-user@ip-172-31-55-79 ~]$ java -jar Interface.jar client -s LOCAL -t 2 -w workload_huge
The total time taken is:1390 ms
[ec2-user@ip-172-31-55-79 ~]$ java -jar Interface.jar client -s LOCAL -t 1 -w workload_huge
The total time taken is:2414 ms
[ec2-user@ip-172-31-55-79 ~]$ java -jar Interface.jar client -s LOCAL -t 8 -w workload_10000ms_80.txt
The total time taken is:100015 ms
[ec2-user@ip-172-31-55-79 ~]$ java -jar Interface.jar client -s LOCAL -t 4 -w workload_10000ms_40.txt
The total time taken is:100012 ms
[ec2-user@ip-172-31-55-79 ~]$ java -jar Interface.jar client -s LOCAL -t 16 -w workload_10000ms_160.txt


ec2-user@ip-172-31-53-50:~
4:04 PM
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.
swathi@swathi-HP-EliteBook-8470p:~$ cd Desktop
swathi@swathi-HP-EliteBook-8470p:~/Desktop$ ssh -i "PA3.pem" ec2-user@ec2-54-152-33-198.compute-1.amazonaws.com
Last login: Sun May 1 17:35:11 2016 from 208-59-159-50.c3-0.mcm-ubr1.chi-mcm.il.cable.rcn.com
Amazon Linux AMI
https://aws.amazon.com/amazon-linux-ami/2016.03-release-notes/
7 package(s) needed for security, out of 23 available
Run "sudo yum update" to apply all updates.
[ec2-user@ip-172-31-53-50 ~]$ java -jar Interface.jar client -s RemoteQ1 -w workload_file.txt
In else block
Here
Done
There
List of tasks ready
The length of jobQueue is 6
Client --- messages put on SQS queue
[ec2-user@ip-172-31-53-50 ~]$ java -jar Interface.jar client -s RemoteQ1 -w jobs_sleep0
List of tasks ready
The length of jobQueue is 10000
Client --- messages put on SQS queue
^C[ec2-user@ip-172-31-53-50 ~]$ java -jar Interface.jar client -s RemoteQ1 -w jobs_sleep0
List of tasks ready
The length of jobQueue is 10000
Client --- messages put on SQS queue


ec2-user@ip-172-31-55-79:~
11:46 PM
[ec2-user@ip-172-31-55-79 ~]$ java -jar Interface.jar client -s LOCAL -t 1 -w workload_10ms_1000.txt
The total time taken is:10210 ms
[ec2-user@ip-172-31-55-79 ~]$ java -jar Interface.jar client -s LOCAL -t 2 -w workload_10ms_2000.txt
The total time taken is:10150 ms
[ec2-user@ip-172-31-55-79 ~]$ java -jar Interface.jar client -s LOCAL -t 4 -w workload_10ms_4000.txt
The total time taken is:10272 ms
[ec2-user@ip-172-31-55-79 ~]$ java -jar Interface.jar client -s LOCAL -t 8 -w workload_10ms_8000.txt
The total time taken is:10142 ms
[ec2-user@ip-172-31-55-79 ~]$ java -jar Interface.jar client -s LOCAL -t 16 -w workload_10ms_16000.txt
The total time taken is:10242 ms
[ec2-user@ip-172-31-55-79 ~]$ java -jar Interface.jar client -s LOCAL -t 16 -w workload_1000ms_100.txt
The total time taken is:7006 ms
[ec2-user@ip-172-31-55-79 ~]$ java -jar Interface.jar client -s LOCAL -t 1 -w workload_1000ms_100.txt
The total time taken is:100029 ms
[ec2-user@ip-172-31-55-79 ~]$ java -jar Interface.jar client -s LOCAL -t 2 -w workload_1000ms_200.txt
The total time taken is:100023 ms
[ec2-user@ip-172-31-55-79 ~]$ java -jar Interface.jar client -s LOCAL -t 4 -w workload_1000ms_400.txt
The total time taken is:100023 ms
[ec2-user@ip-172-31-55-79 ~]$ java -jar Interface.jar client -s LOCAL -t 8 -w workload_1000ms_800.txt
The total time taken is:100031 ms
[ec2-user@ip-172-31-55-79 ~]$ java -jar Interface.jar client -s LOCAL -t 16 -w workload_1000ms_1600.txt
The total time taken is:100052 ms
[ec2-user@ip-172-31-55-79 ~]$ java -jar Interface.jar client -s LOCAL -t 1 -w workload_10000ms_10.txt
The total time taken is:100012 ms
[ec2-user@ip-172-31-55-79 ~]$ java -jar Interface.jar client -s LOCAL -t 2 -w workload_10000ms_20.txt
The total time taken is:100009 ms
[ec2-user@ip-172-31-55-79 ~]$ java -jar Interface.jar client -s LOCAL -t 1 -w workload_huge
The total time taken is:1159 ms
[ec2-user@ip-172-31-55-79 ~]$ java -jar Interface.jar client -s LOCAL -t 2 -w workload_huge
The total time taken is:1390 ms
[ec2-user@ip-172-31-55-79 ~]$ java -jar Interface.jar client -s LOCAL -t 1 -w workload_huge
The total time taken is:2414 ms
[ec2-user@ip-172-31-55-79 ~]$ java -jar Interface.jar client -s LOCAL -t 8 -w workload_10000ms_80.txt
The total time taken is:100015 ms
[ec2-user@ip-172-31-55-79 ~]$ java -jar Interface.jar client -s LOCAL -t 4 -w workload_10000ms_40.txt
The total time taken is:100012 ms
[ec2-user@ip-172-31-55-79 ~]$ java -jar Interface.jar client -s LOCAL -t 16 -w workload_10000ms_160.txt
The total time taken is:100018 ms
[ec2-user@ip-172-31-55-79 ~]$ java -jar Interface.jar client -s LOCAL -t 1 -w workload_huge
The total time taken is:29082 ms
[ec2-user@ip-172-31-55-79 ~]$ java -jar Interface.jar client -s LOCAL -t 2 -w workload_huge


ec2-user@ip-172-31-53-50:~
3:19 AM
Before the wait loop
Out of the wait loop
The total time taken is 41421
[ec2-user@ip-172-31-53-50 ~]$ java -jar Interface.jar client -s RemoteQ -w workload_1000ms_200.txt
Client --- messages put on SQS queue
Before the wait loop
Out of the wait loop
The total time taken is 1143932
[ec2-user@ip-172-31-53-50 ~]$ java -jar Interface.jar client -s RemoteQ -w workload_1000ms_400.txt
Client --- messages put on SQS queue
Before the wait loop
Out of the wait loop
The total time taken is 29892
[ec2-user@ip-172-31-53-50 ~]$ java -jar Interface.jar client -s RemoteQ -w workload_1000ms_800.txt
Client --- messages put on SQS queue
FileZilla
Out of the wait loop
The total time taken is 576054
[ec2-user@ip-172-31-53-50 ~]$ java -jar Interface.jar client -s RemoteQ -w workload_1000ms_400.txt
Client --- messages put on SQS queue
Before the wait loop
Out of the wait loop
The total time taken is 573416
[ec2-user@ip-172-31-53-50 ~]$ java -jar Interface.jar client -s RemoteQ -w workload_1000ms_200.txt
Client --- messages put on SQS queue
Before the wait loop
Out of the wait loop
The total time taken is 573750
[ec2-user@ip-172-31-53-50 ~]$ java -jar Interface.jar client -s RemoteQ -w workload_1000ms_1600.txt
Client --- messages put on SQS queue
Before the wait loop
Out of the wait loop
The total time taken is 621291
[ec2-user@ip-172-31-53-50 ~]$
[ec2-user@ip-172-31-53-50 ~]$ java -jar Interface.jar client -s RemoteQ -w workload_10000ms_10.txt
Client --- messages put on SQS queue
Before the wait loop
Out of the wait loop
The total time taken is 116666
[ec2-user@ip-172-31-53-50 ~]$ java -jar Interface.jar client -s RemoteQ -w workload_10000ms_20.txt

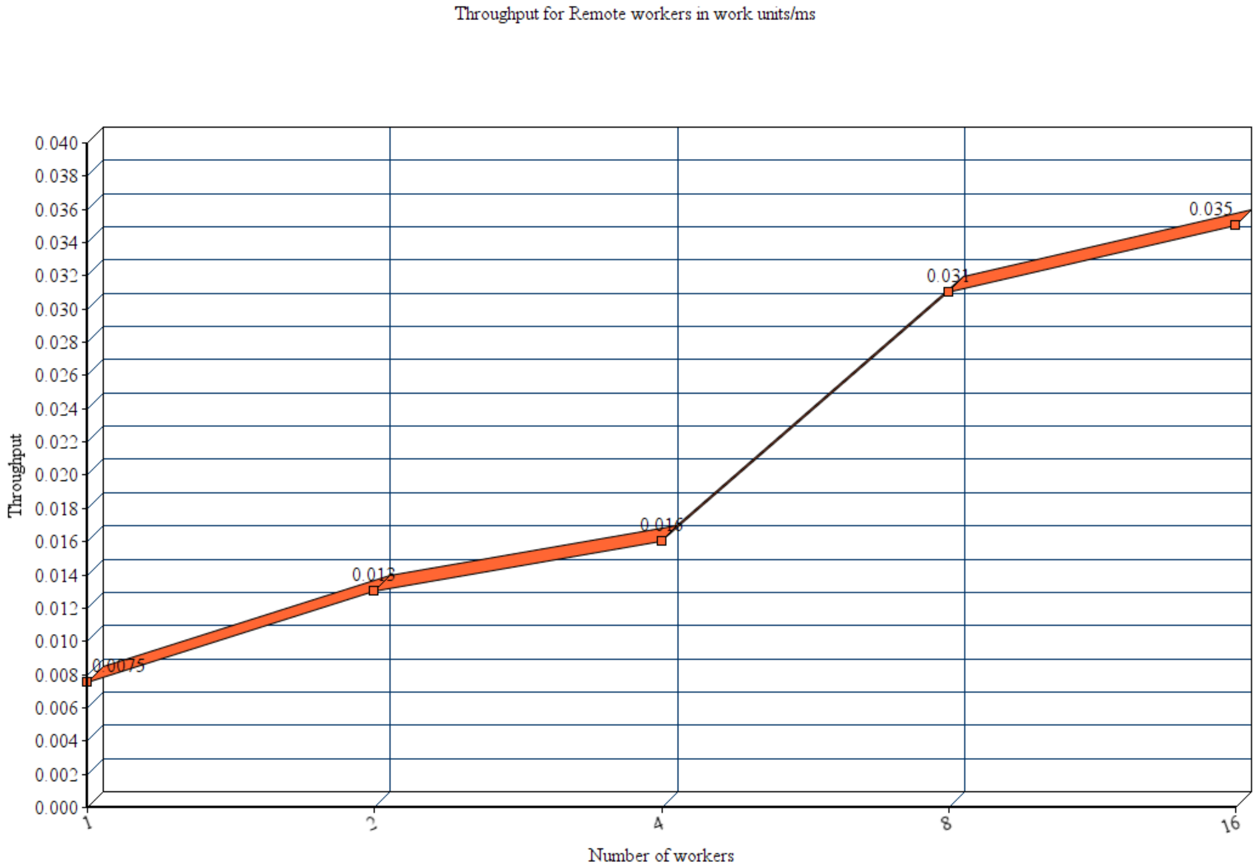
```
ec2-user@ip-172-31-53-50:~
Before the wait loop
Out of the wait loop
The total time taken is 573416
[ec2-user@ip-172-31-53-50 ~]$ java -jar Interface.jar client -s RemoteQ -w workload_1000ms_200.txt
Client --- messages put on SQS queue
Before the wait loop
Out of the wait loop
The total time taken is 573750
[ec2-user@ip-172-31-53-50 ~]$ java -jar Interface.jar client -s RemoteQ -w workload_1000ms_1600.txt

Client --- messages put on SQS queue
Before the wait loop
Out of the wait loop
The total time taken is 621291
[ec2-user@ip-172-31-53-50 ~]$
[ec2-user@ip-172-31-53-50 ~]$ java -jar Interface.jar client -s RemoteQ -w workload_10000ms_10.txt
Client --- messages put on SQS queue
Before the wait loop
Out of the wait loop
The total time taken is 116666
[ec2-user@ip-172-31-53-50 ~]$ java -jar Interface.jar client -s RemoteQ -w workload_10000ms_20.txt
Client --- messages put on SQS queue
Before the wait loop
Out of the wait loop
The total time taken is 106218
[ec2-user@ip-172-31-53-50 ~]$ java -jar Interface.jar client -s RemoteQ -w workload_10000ms_40.txt
Client --- messages put on SQS queue
Before the wait loop
Out of the wait loop
The total time taken is 111754
[ec2-user@ip-172-31-53-50 ~]$ java -jar Interface.jar client -s RemoteQ -w workload_10000ms_80.txt
Client --- messages put on SQS queue
Before the wait loop
Out of the wait loop
The total time taken is 1339
[ec2-user@ip-172-31-53-50 ~]$ java -jar Interface.jar client -s RemoteQ -w workload_10000ms_80.txt
Client --- messages put on SQS queue
Before the wait loop
Out of the wait loop
The total time taken is 187317
[ec2-user@ip-172-31-53-50 ~]$
```

```
ec2-user@ip-172-31-53-50:~
ec2-user@ip-172-31-53-50:~
The total time taken is 573750
[ec2-user@ip-172-31-53-50 ~]$ java -jar Interface.jar client -s RemoteQ -w workload_1000ms_1600.txt

Client --- messages put on SQS queue
Before the wait loop
Out of the wait loop
The total time taken is 621291
[ec2-user@ip-172-31-53-50 ~]$
[ec2-user@ip-172-31-53-50 ~]$ java -jar Interface.jar client -s RemoteQ -w workload_10000ms_10.txt
Client --- messages put on SQS queue
Before the wait loop
Out of the wait loop
The total time taken is 116666
[ec2-user@ip-172-31-53-50 ~]$ java -jar Interface.jar client -s RemoteQ -w workload_10000ms_20.txt
Client --- messages put on SQS queue
Before the wait loop
Out of the wait loop
The total time taken is 106218
[ec2-user@ip-172-31-53-50 ~]$ java -jar Interface.jar client -s RemoteQ -w workload_10000ms_40.txt
Client --- messages put on SQS queue
Before the wait loop
Out of the wait loop
The total time taken is 111754
[ec2-user@ip-172-31-53-50 ~]$ java -jar Interface.jar client -s RemoteQ -w workload_10000ms_80.txt
Client --- messages put on SQS queue
Before the wait loop
Out of the wait loop
The total time taken is 1339
[ec2-user@ip-172-31-53-50 ~]$ java -jar Interface.jar client -s RemoteQ -w workload_10000ms_80.txt
Client --- messages put on SQS queue
Before the wait loop
Out of the wait loop
The total time taken is 187317
[ec2-user@ip-172-31-53-50 ~]$ java -jar Interface.jar client -s RemoteQ -w workload_10000ms_160.txt
Client --- messages put on SQS queue
Before the wait loop
Out of the wait loop
The total time taken is 168075
[ec2-user@ip-172-31-53-50 ~]$
```

PERFORMANCE EVALUATION:

Throughput:

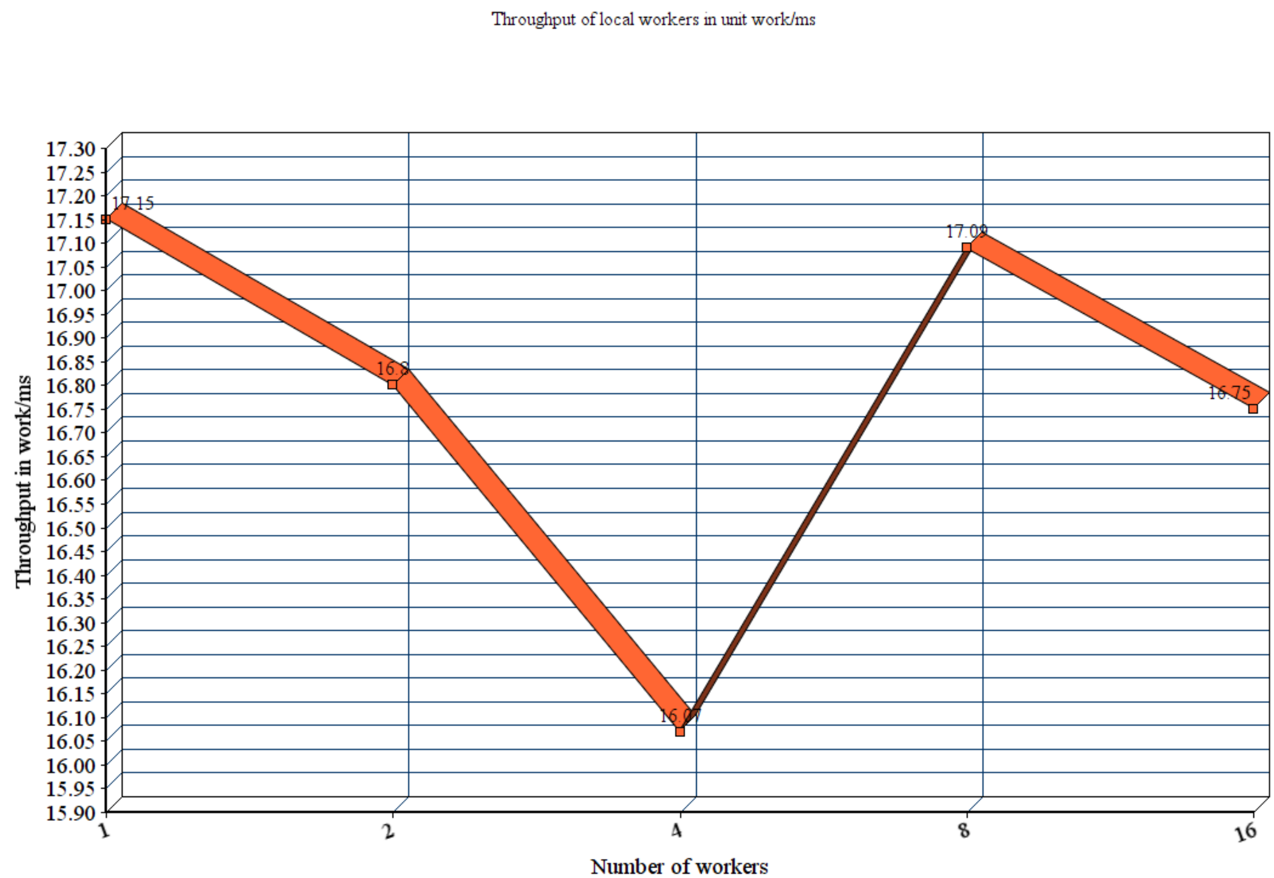
Remote Worker:



We notice that as the number of workers scales, the throughput increases. This is because the workers don't block on executing a task and rather pick up tasks when they are free to run them. Here we also need to consider latency due to communication over network and interaction with DynamoDB.

Number of workers	Throughput in units/ms
1	0.0075
2	0.013
4	0.016
8	0.031
16	0.035

Local worker:



The throughput for local worker shows a sharp fluctuation in the performance. This is because the operation is done via in-memory queue and it depends on the burst of resources available at the moment to run the job. There is only the memory constraint to deal with which causes a little delay but still compared to remote workers, it is faster.

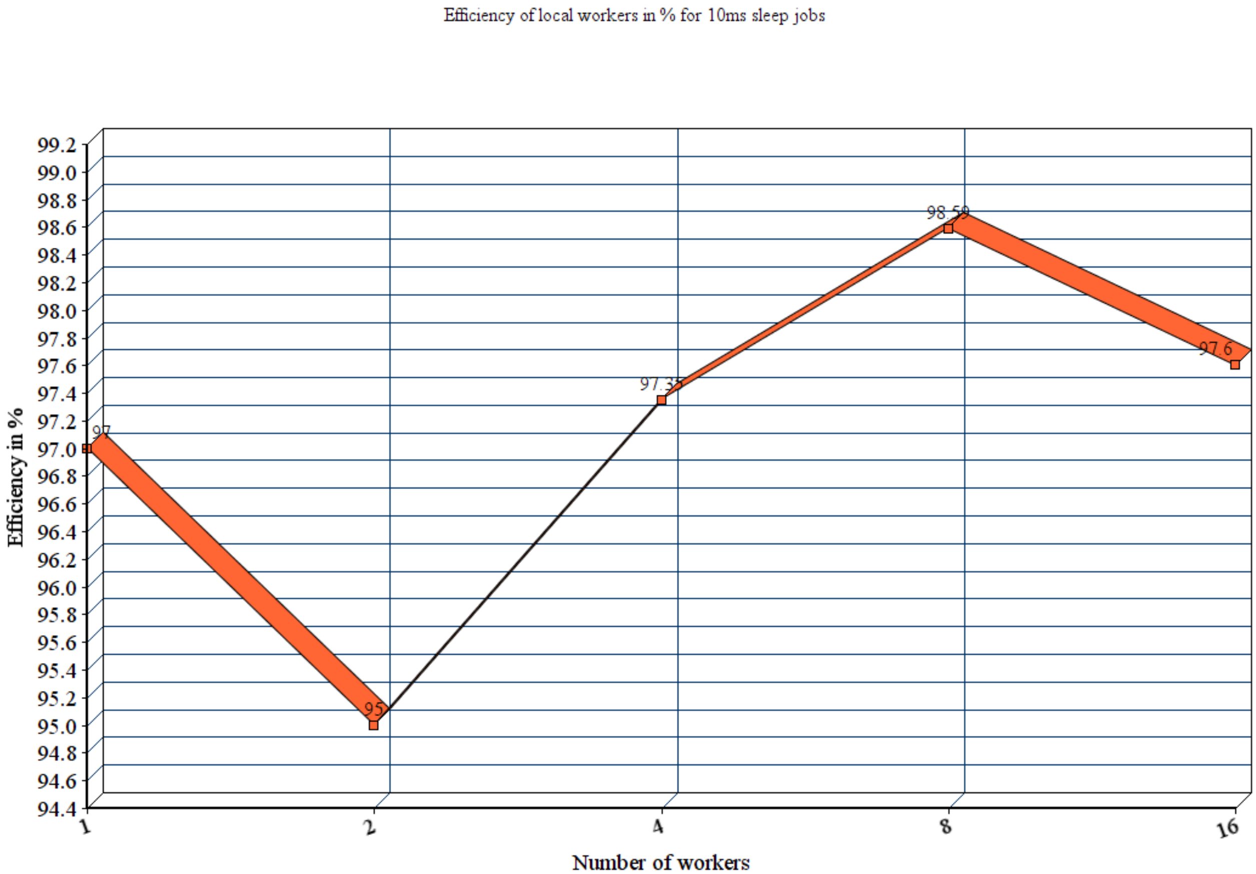
Number of workers	Throughput
1	17.15
2	16.8
4	16.07
8	17.09
16	16.75

Efficiency:

Local worker:

The efficiency is calculated as the ratio of ideal time by measured time.

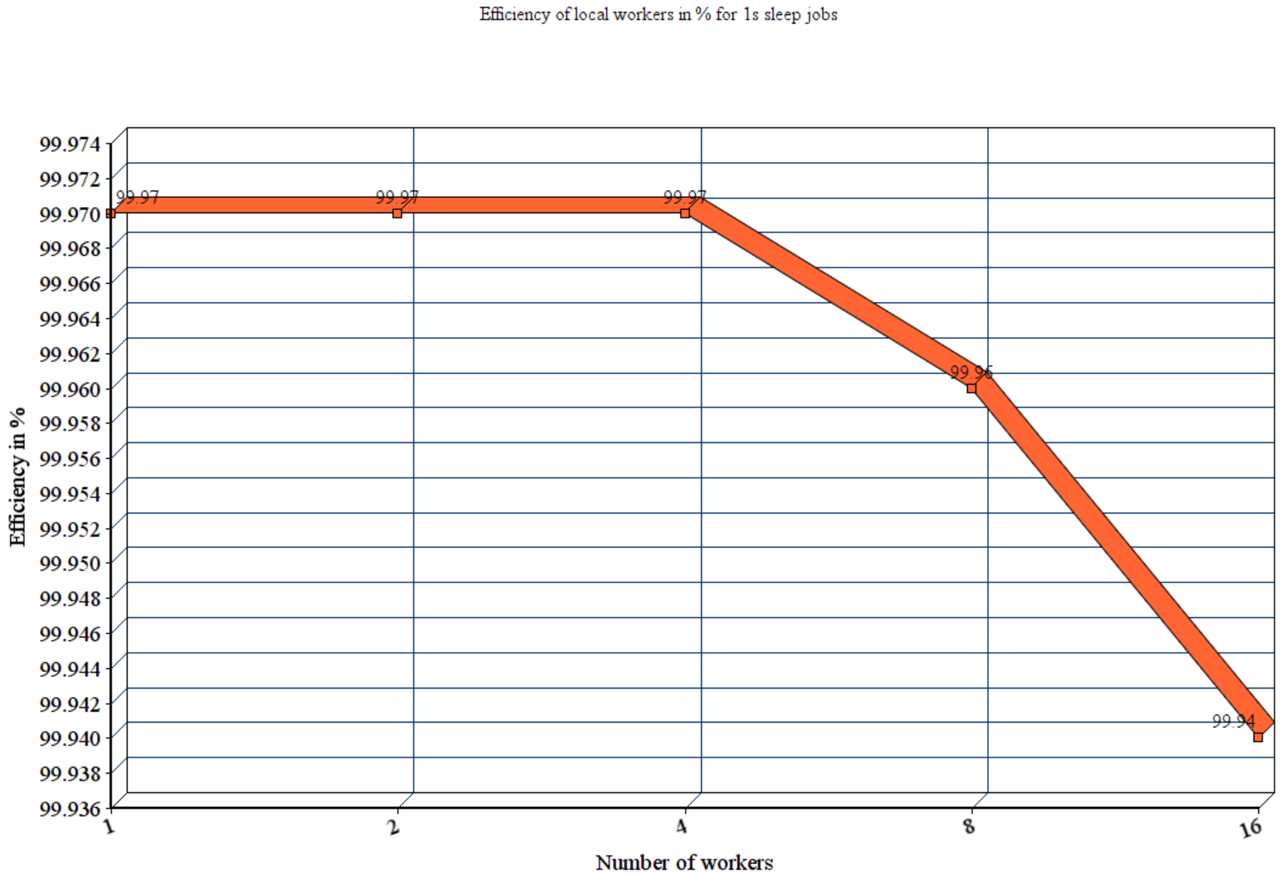
10 ms:



From the plot we observe that the efficiency rises steadily and as the scale increases it improves. This is due to the design that there is no delay in waiting for jobs due to queuing delay or DB operations. The workers pick up the jobs from the in-memory queue and which is faster and more efficient.

Number of workers	Efficiency
1	97%
2	95%
4	97.35%
8	98.59%
16	97.6%

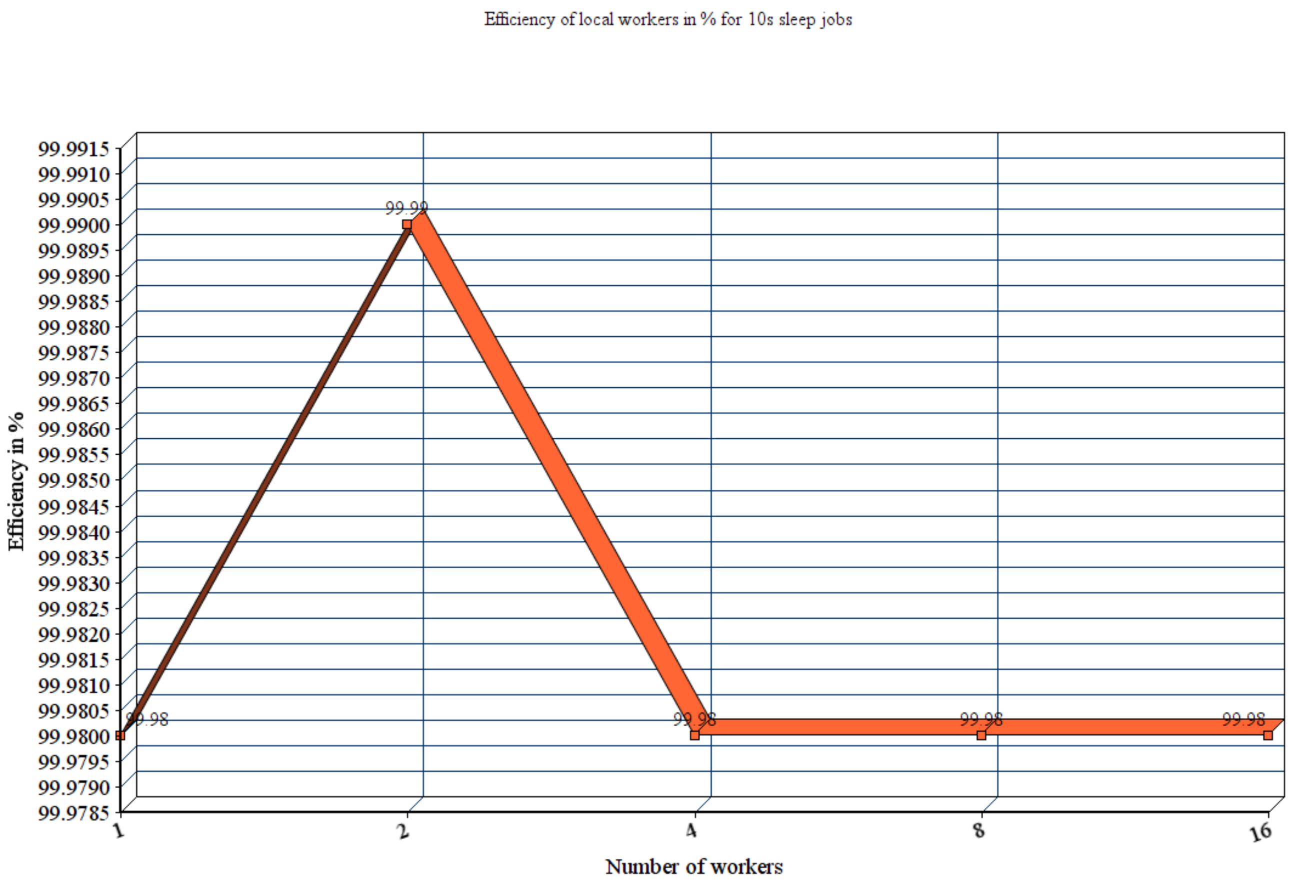
1s:



There is a slight drop in efficiency for 1s sleep jobs because, when the worker sleeps for 1000ms or 1s, there is extra delay in addition to queuing delay and may lead to some contention among workers for the jobs. Duplicate jobs may get picked and dropped thus hampering efficiency.

Number of workers	Efficiency
1	99.97%
2	99.97%
4	99.97%
8	99.96%
16	99.94%

10s:

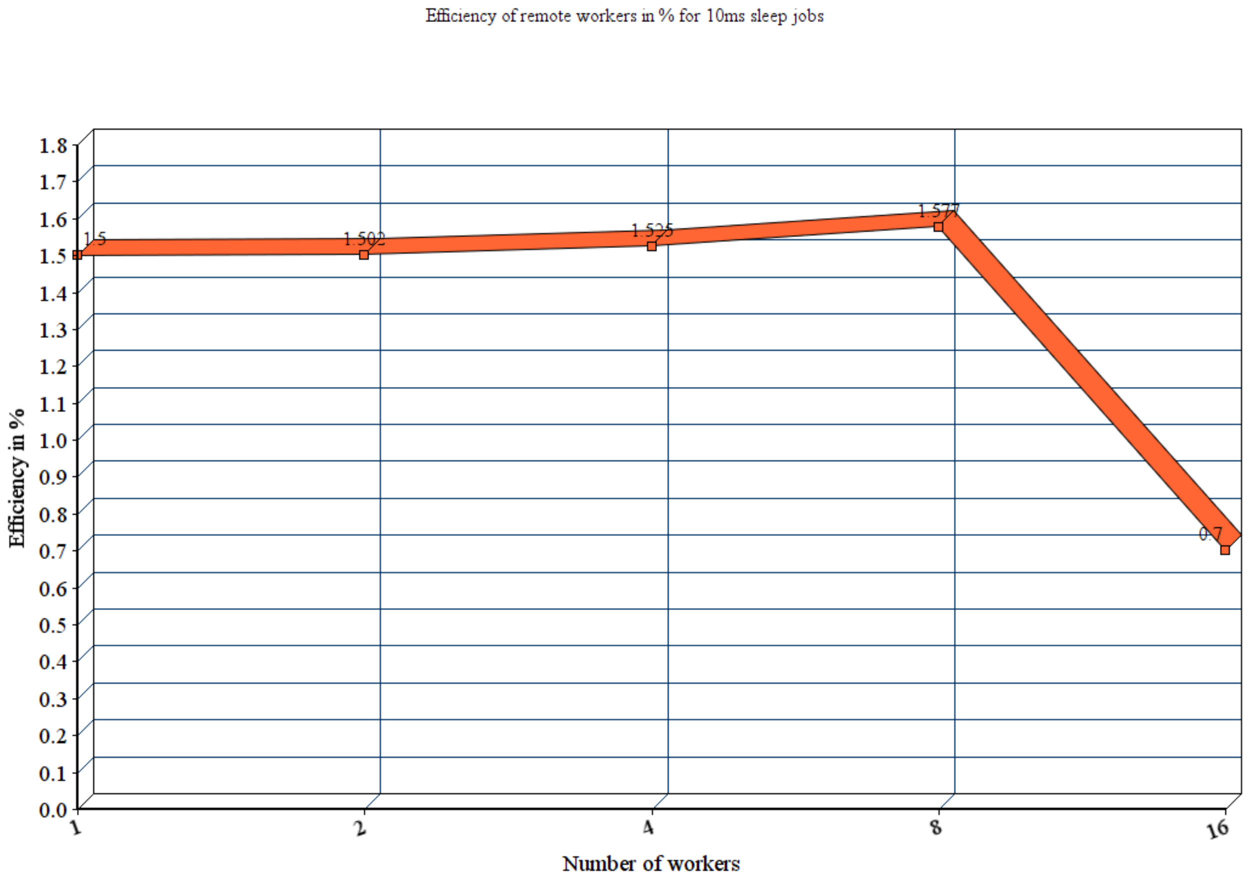


The 10s sleep jobs slightly increases for 2 workers but is back to stable value for other workers. They scale well for higher value of workers due to interaction via in-memory queue and maintain a steady and consistent efficiency.

Number of workers	Efficiency
1	99.98%
2	99.99%
4	99.98%
8	99.98%
16	99.98%

Remote Workers:

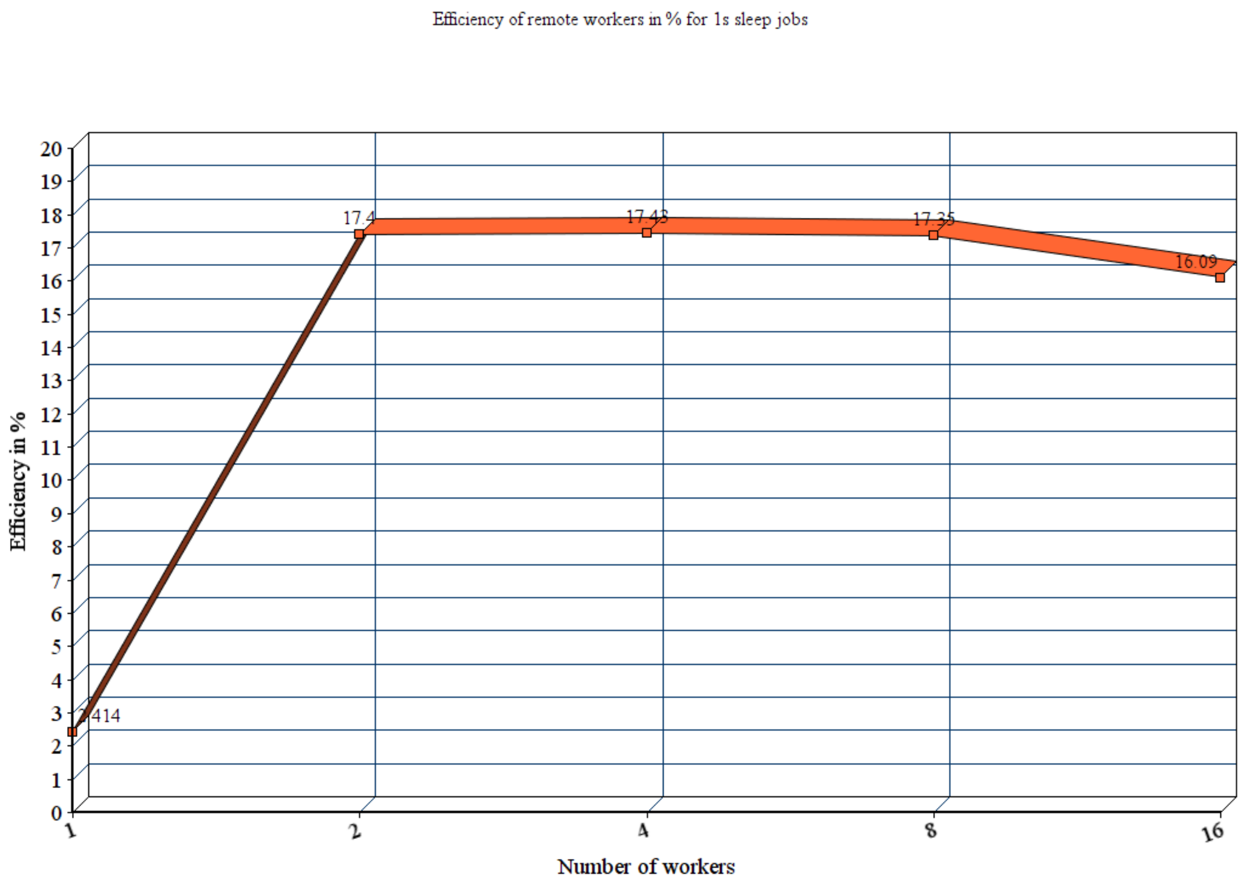
10 ms:



The values are steadily increasing and suddenly decrease for number of workers = 16. This happens because of lot of overhead in managing the operations such as delete and write and read of response queue for 16 workers at the same time. There is imperfect load balance as number of nodes increases.

Number of workers	Efficiency
1	1.5%
2	1.502%
4	1.525%
8	1.577%
16	0.7%

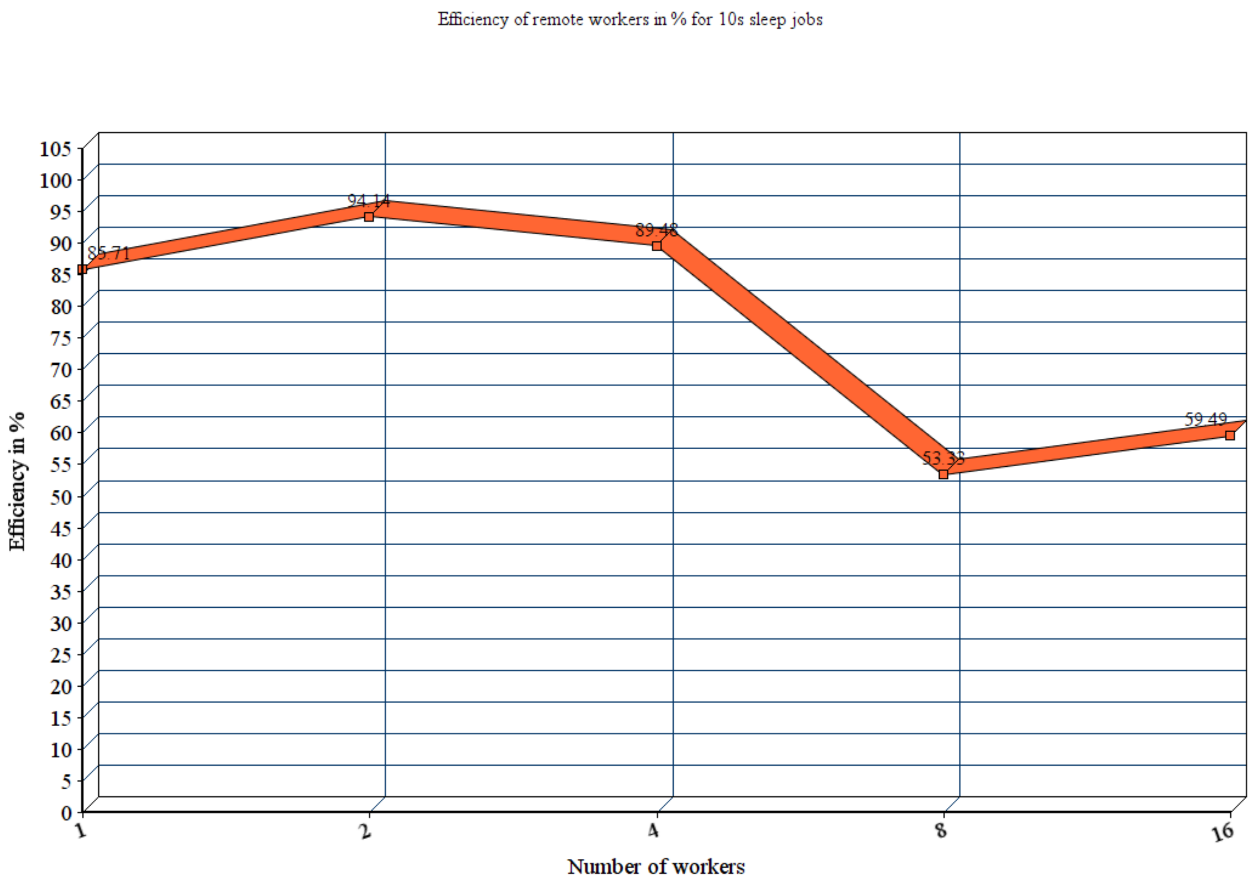
1s:



The above plot shows that the remote workers scale well and slightly reduce towards the end. They are efficient as the sleep time is sufficient enough for them to poll and pick the task without being idle or blocked for long time. There is imperfect load balance as number of nodes increases.

Number of workers	Efficiency
1	2.414%
2	17.40%
4	17.43%
8	17.35%
16	16.09%

10s:



From the plot we observe that there is a sudden drop after 8 workers and then it again increases steadily for sleep time = 10s. This is because more delay is incurred when the worker itself does a sleep operation. There are more chances of resource contention like duplicate tasks and in general a delay in communication. There is imperfect load balance as number of nodes increases.

Number of workers	Efficiency
1	85.71%
2	94.14%
4	84.98%
8	53.33%
16	59.49%

References:

[1] J. ThreadPoolExecutor and +. Kumar, "Java Thread Pool Example using Executors and ThreadPoolExecutor", *Java Code Geeks*, 2016. [Online]. Available: https://www.javacodegeeks.com/2013/01/java-thread-pool-example-using-executors-and-threadpoolexecutor.html. [Accessed: 04- May- 2016].

[2] "How to use CloudKon - Iman Sadooghi", *Sites.google.com*, 2016. [Online]. Available: https://sites.google.com/site/imansadooghi/cloudkon/how-to-use-cloudkon. [Accessed: 04- May- 2016].

[9] aws/aws-sdk-java", *GitHub*, 2016. [Online]. Available: https://github.com/aws/aws-sdk-java/tree/master/src/samples. [Accessed: 04- May- 2016].