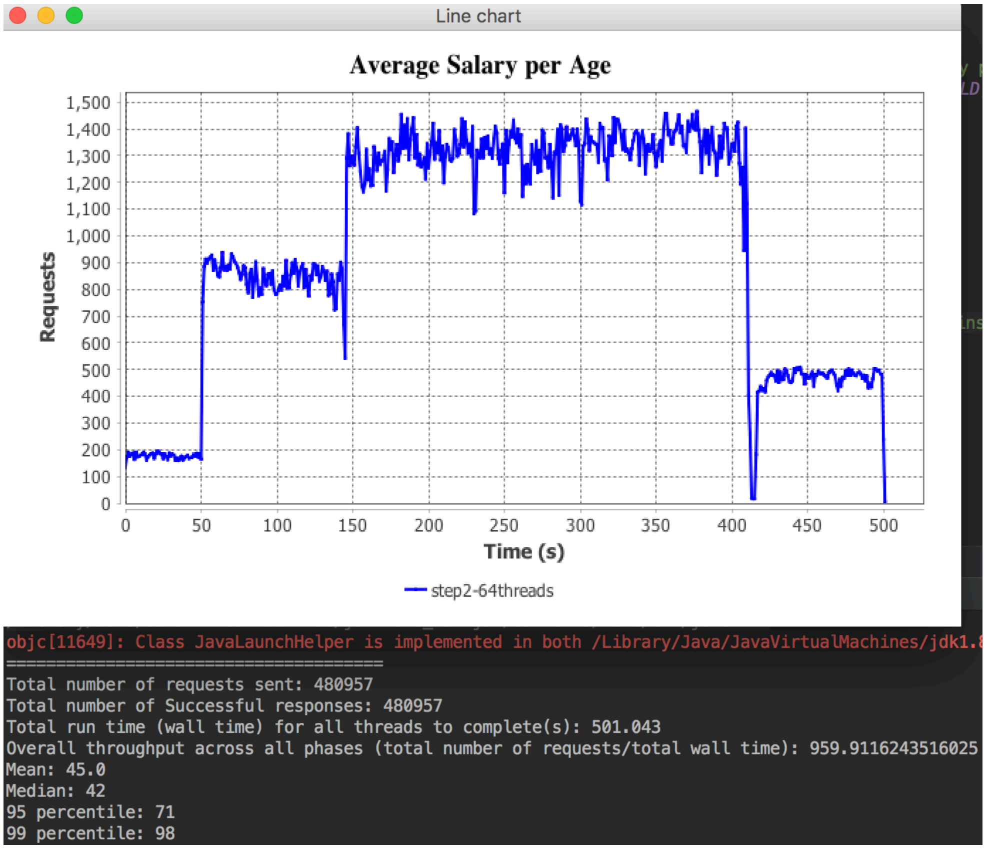
Assignment2. Tao Xu

Step1 and 2:

64 threads and 100 iterations

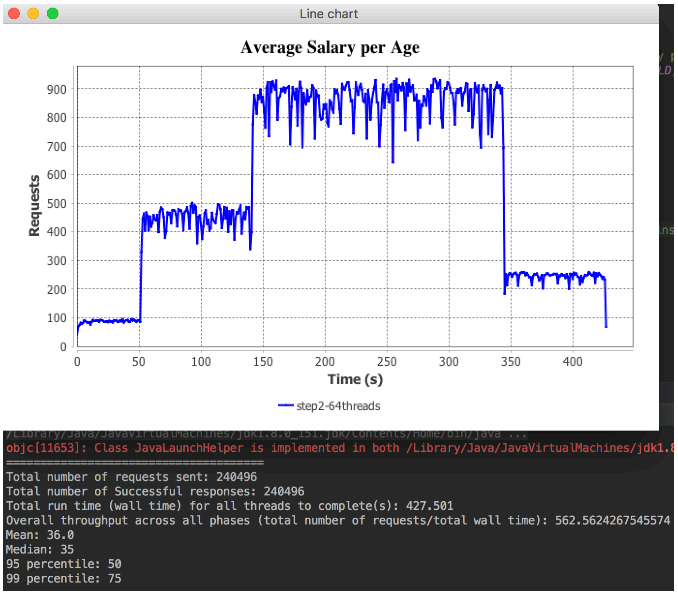


```
objc[11649]: Class JavaLaunchHelper is implemented in both /Library/Java/JavaVirtualMachines/jdk1.8
=====================================
Total number of requests sent: 480957
Total number of Successful responses: 480957
Total run time (wall time) for all threads to complete(s): 501.043
Overall throughput across all phases (total number of requests/total wall time): 959.9116243516025
Mean: 45.0
Median: 42
95 percentile: 71
99 percentile: 98
```

Comments: 8 mins for a 64-threads test seems very straight forward.

Step3:

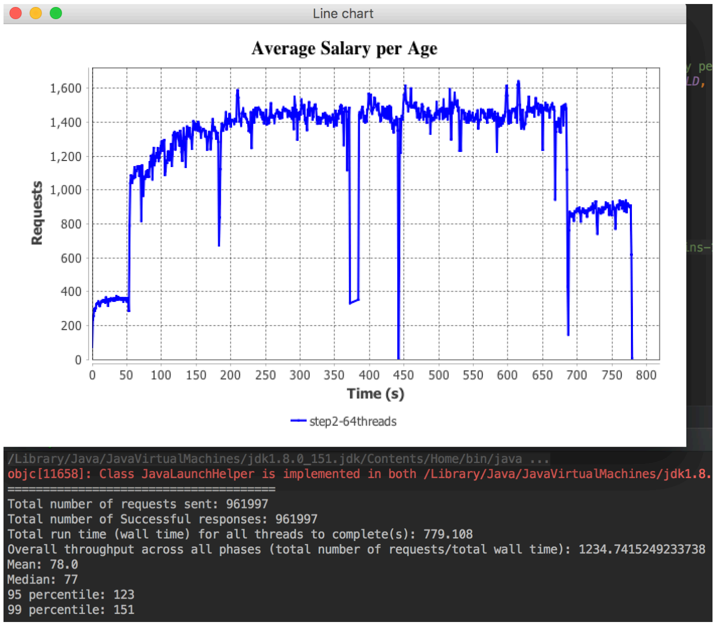
32 threads:



```
/Library/Java/JavaVirtualMachines/jdk1.8.0_151.jdk/Contents/Home/bin/java ...
objc[11653]: Class JavaLaunchHelper is implemented in both /Library/Java/JavaVirtualMachines/jdk1.8
=====================================
Total number of requests sent: 240496
Total number of Successful responses: 240496
Total run time (wall time) for all threads to complete(s): 427.501
Overall throughput across all phases (total number of requests/total wall time): 562.5624267545574
Mean: 36.0
Median: 35
95 percentile: 50
99 percentile: 75
```

Comments: 32-threads have less latency, less throughput and shorter test period.

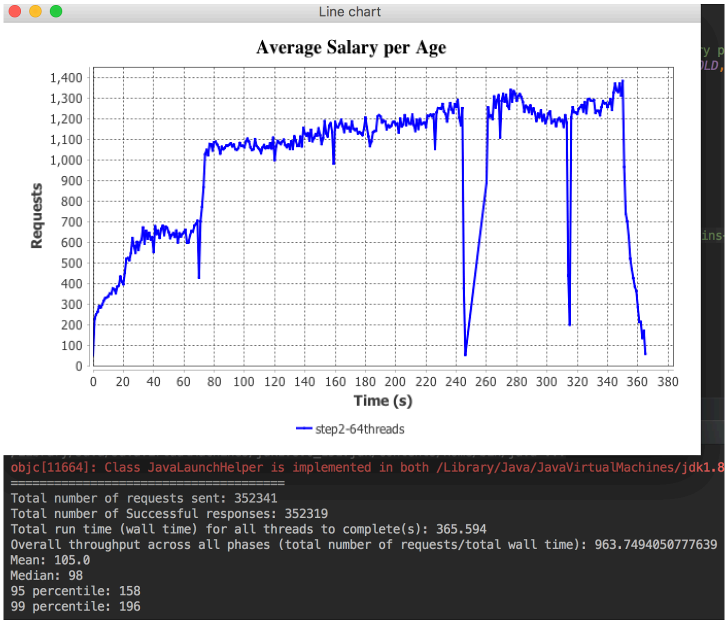
64 threads: (as above in step2)
128 threads:



```
/Library/Java/JavaVirtualMachines/jdk1.8.0_151.jdk/Contents/Home/bin/java ...
objc[11658]: Class JavaLaunchHelper is implemented in both /Library/Java/JavaVirtualMachines/jdk1.8.
======================================
Total number of requests sent: 961997
Total number of Successful responses: 961997
Total run time (wall time) for all threads to complete(s): 779.108
Overall throughput across all phases (total number of requests/total wall time): 1234.7415249233738
Mean: 78.0
Median: 77
95 percentile: 123
99 percentile: 151
```

Comments: larger latency but still acceptable, throughputs as large as 1300/s

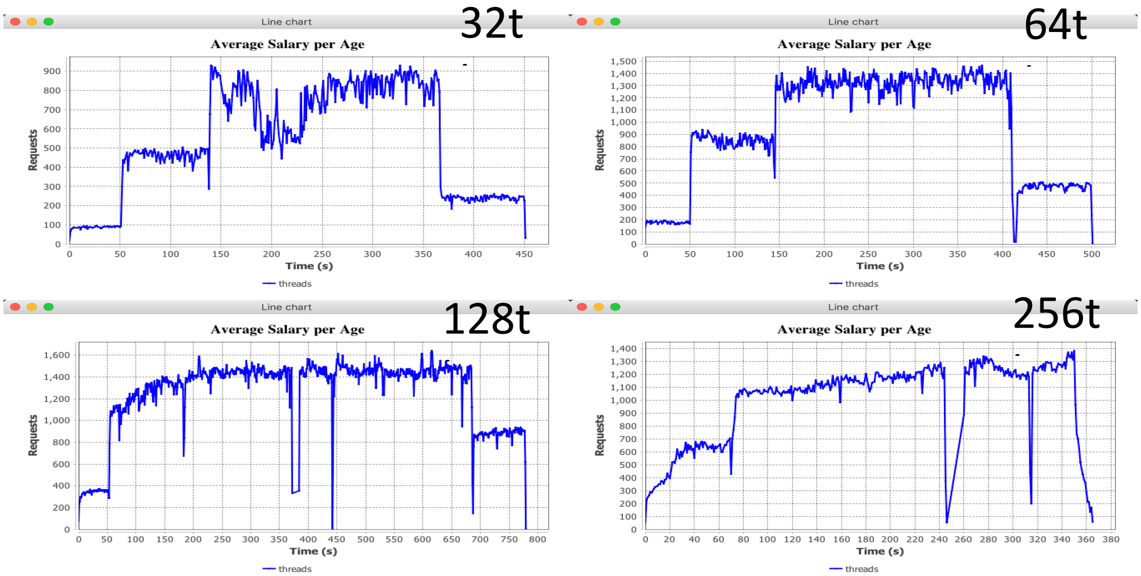
256 threads:



```
objc[11664]: Class JavaLaunchHelper is implemented in both /Library/Java/JavaVirtualMachines/jdk1.8
======================================
Total number of requests sent: 352341
Total number of Successful responses: 352319
Total run time (wall time) for all threads to complete(s): 365.594
Overall throughput across all phases (total number of requests/total wall time): 963.7494050777639
Mean: 105.0
Median: 98
95 percentile: 158
99 percentile: 196
```

Comments: 256-threads test becomes slow as the throughput reach the ceiling, the latency become large and the figure shape is screwed. Also 0.001% requests start to fail. This indicate the single sever system has reached its limit.

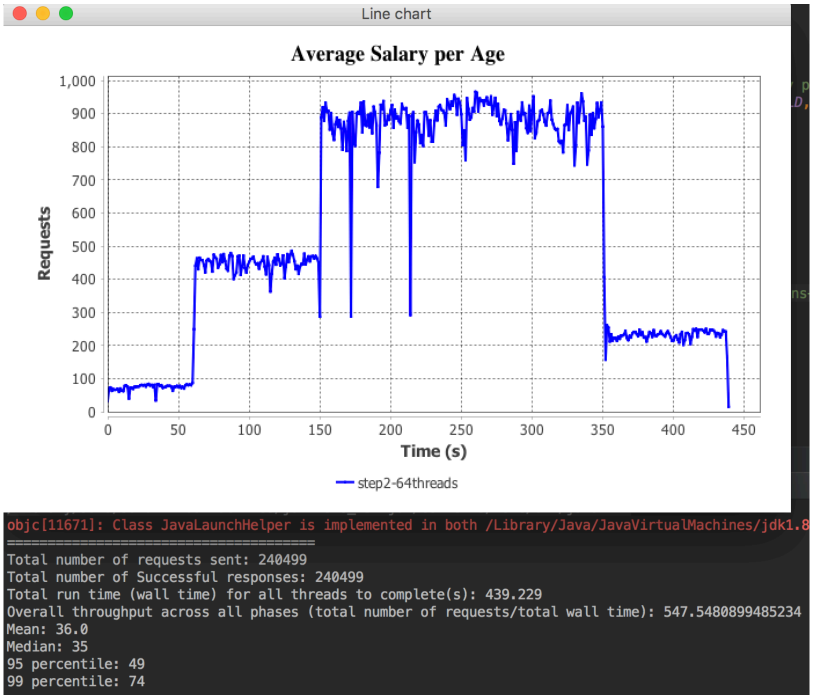
Single Server test in the same picture:



Comments: this indicates the performance difference under different thread loads.

Step4:
Load Balancer Set up: 4 ec2 instance are grouped together into the load balancer. High CPU utilization, high memory usage and failed requests will all trigger the balancing rule.

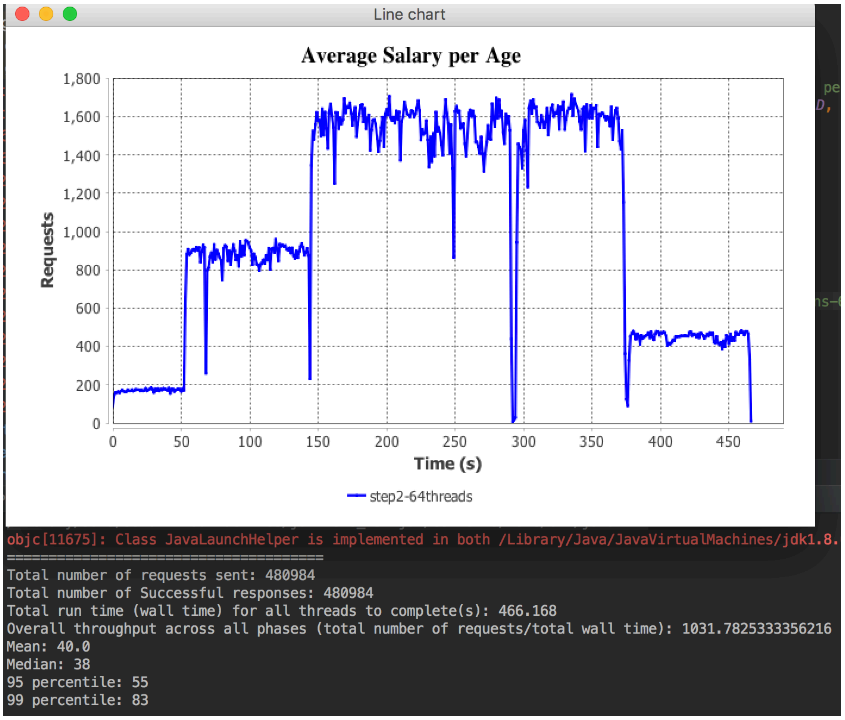
32 threads:



```
objc[11671]: Class JavaLaunchHelper is implemented in both /Library/Java/JavaVirtualMachines/jdk1.8
=====================================
Total number of requests sent: 240499
Total number of Successful responses: 240499
Total run time (wall time) for all threads to complete(s): 439.229
Overall throughput across all phases (total number of requests/total wall time): 547.5480899485234
Mean: 36.0
Median: 35
95 percentile: 49
99 percentile: 74
```

Comments: compare to single sever, it only improves a little, because it hasn’t trigger load balance rule.

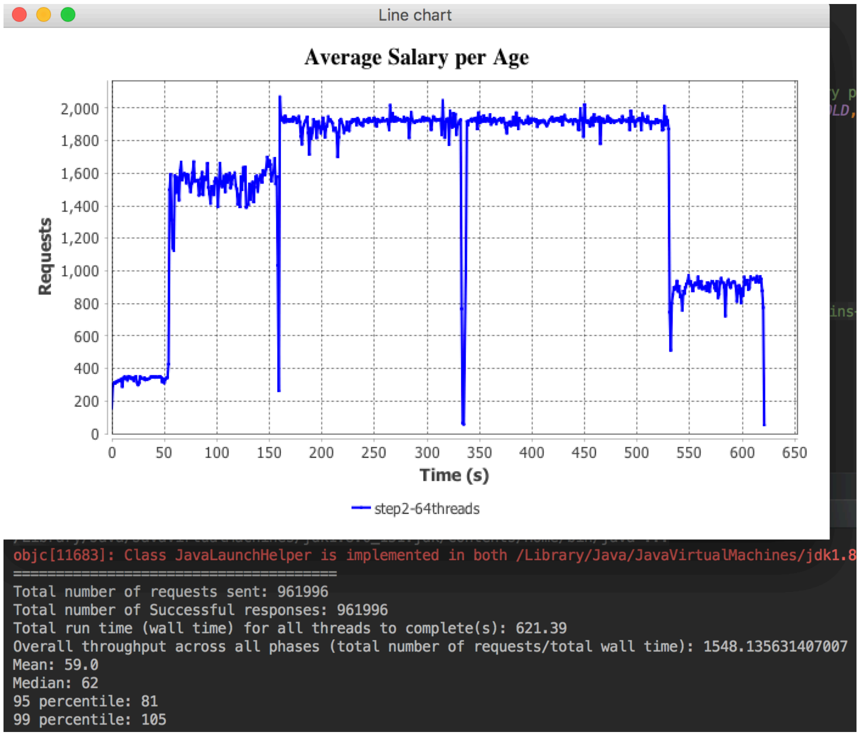
64 threads:



```
objc[11675]: Class JavaLaunchHelper is implemented in both /Library/Java/JavaVirtualMachines/jdk1.8.
====================================
Total number of requests sent: 480984
Total number of Successful responses: 480984
Total run time (wall time) for all threads to complete(s): 466.168
Overall throughput across all phases (total number of requests/total wall time): 1031.7825333356216
Mean: 40.0
Median: 38
95 percentile: 55
99 percentile: 83
```

Comment: Compare to single sever, it noticeably reduces the latency and total wall time, also the throughput gets larger. This shows the load balancer has started to join in.

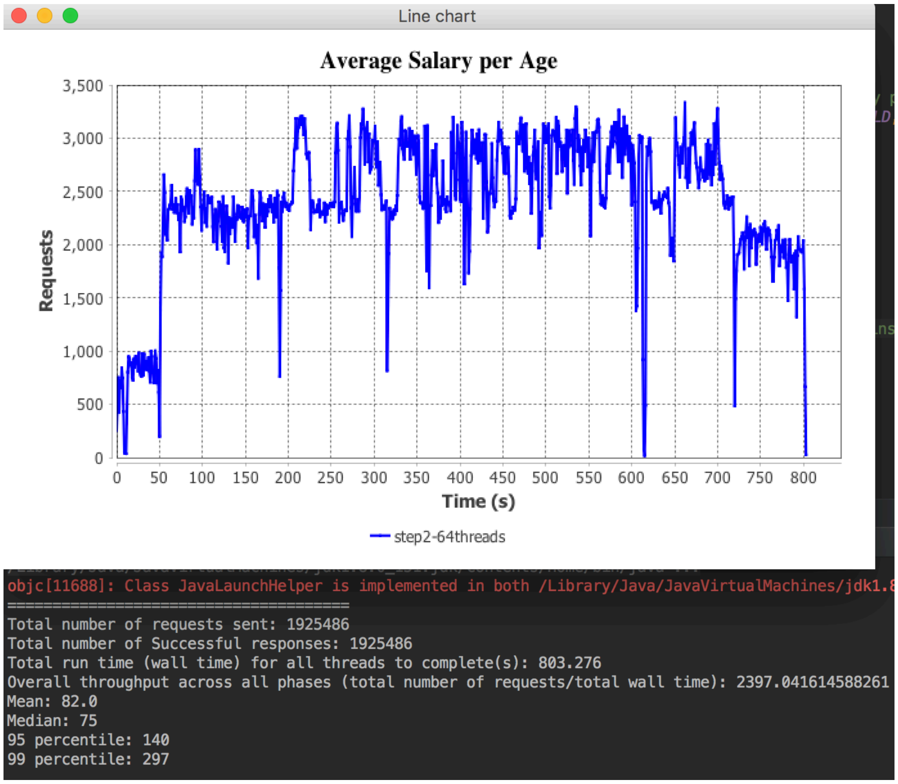
128 threads:



```
objc[11683]: Class JavaLaunchHelper is implemented in both /Library/Java/JavaVirtualMachines/jdk1.8
====================================
Total number of requests sent: 961996
Total number of Successful responses: 961996
Total run time (wall time) for all threads to complete(s): 621.39
Overall throughput across all phases (total number of requests/total wall time): 1548.135631407007
Mean: 59.0
Median: 62
95 percentile: 81
99 percentile: 105
```

Comments: Compare to the single sever system, it improves a lot, here the load balancer work very well to improve the whole performance.

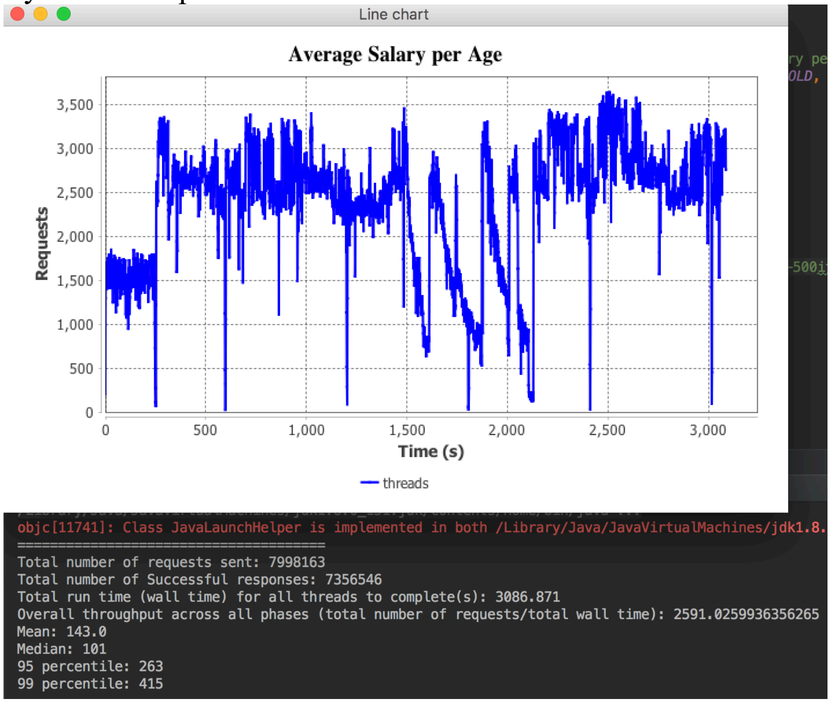
256 threads:



```
objc[11688]: Class JavaLaunchHelper is implemented in both /Library/Java/JavaVirtualMachines/jdk1.8
=====================================
Total number of requests sent: 1925486
Total number of Successful responses: 1925486
Total run time (wall time) for all threads to complete(s): 803.276
Overall throughput across all phases (total number of requests/total wall time): 2397.041614588261
Mean: 82.0
Median: 75
95 percentile: 140
99 percentile: 297
```

Comments: The improvement is obvious from the single server, the graph becomes a good shape and the latency distribution and throughput is very desirable.

Step5 Bonus: My best attempt: 512 threads and 500 iterations:



```
objc[11741]: Class JavaLaunchHelper is implemented in both /Library/Java/JavaVirtualMachines/jdk1.8.
=====================================
Total number of requests sent: 7998163
Total number of Successful responses: 7356546
Total run time (wall time) for all threads to complete(s): 3086.871
Overall throughput across all phases (total number of requests/total wall time): 2591.0259936356265
Mean: 143.0
Median: 101
95 percentile: 263
99 percentile: 415
```

Comments: the throughput is satisfactory but the latency is also very high. This reaches the limit of the whole system.