Wearable Device Data Management- Assignment2

Wen Zhang

Step1 & 2: Server and Client implementation, Performance Analysis of running with default client setting:

Answer:

Implementation:

1. Server:
    - HTTP Requests handlers are implemented in JAX-RS.
    - The server-database communication is implemented by JDBC and C3P0 connection pool.
    - Server is deployed to AWS EC2 instance using tomcat server.
    - DataBase server: AWS RDS MySQL Medium tier.
2. Client:
    - Option Class: to store the clients' requirements from the console.
    - OptionParser: to parse the console input arguments and store the values into an Option Object.
    - Worker Class: to implement Runnable, which is responsible to implement the execution of each thread.
    - JfreeChart Class: to plot the final throughput chart.
    - WDDMClient Class: main thread, to execute four phases based on input configuration, and generate system performance analysis.
    - Details: Client application saves every latency and timestamp into an output file for later performance evaluation.
    - Four phases in my implementation are Non-overlapping, and I use count down latch for execution synchronization.

The plot and performance statistics:

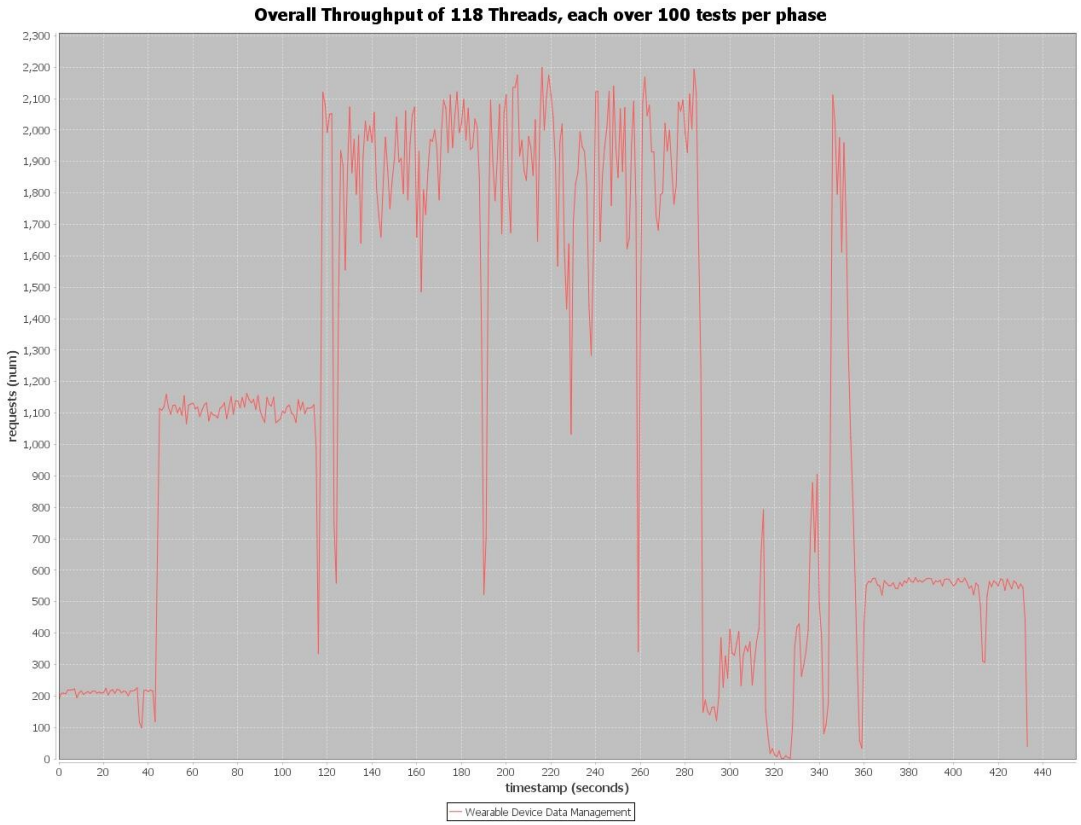
Performance Statistics:

```
Markers  Properties  Servers  Data Source Explorer  Snippets  Console  Search
<terminated> WDDMClient [Java Application] C:\Program Files\Java\jre1.8.0_181\bin\javaw.exe (Nov 3, 2018, 11:01:50 PM)
url: http://34.218.227.254:8080/homework2/webapi/hw2server
Client starting.....Time: 2018/11/03 23:01:50
Warmup phase: All threads running.....
Warmup phase complete: Time 43.767 seconds
Loading phase: All threads running.....
Loading phase complete: Time 73.129 seconds
Peak phase: All threads running.....
Peak phase complete: Time 243.152 seconds
Cooldown phase: All threads running.....
Cooldown phase complete: Time 73.343 seconds
Client Execution Complete.

=====================Execution Summary=========================
Generating XYLineChart for the overall throughput...
XYLineChart WDDMThroughput.jpeg generated successfully.
Test Wall Time: 433.401 seconds
Total number of request sent: 481000
Total number of Successful responses: 481000
Overall throughput across all phases: 1,109.827 requests per second
Mean latencies for all requests: 39 MilliSeconds*
Median latencies for all requests: 29 MilliSeconds*
95th percentile latency for all requests: 51 MilliSeconds*
99th percentile latency for all requests: 319 MilliSeconds*
```

*Default client setting: **MaxThreads: 64**; Day number:1; User population: 100,000; Number of tests per phase: 100.*

XY-LineChart of overall throughput:



Note: The thread number in the above chart is the total number of threads generated during all four Phases (e.g: 6 + 32 + 64 + 16 = 118).

Step3: The plot and performance statistics of four different maxThreads values:

1. Performance Statistics: maxThread = 32

```
Markers  Properties  Servers  Data Source Explorer  Snippets  Console  Search
<terminated> WDDMClient [Java Application] C:\Program Files\Java\jre1.8.0_181\bin\javaw.exe (Nov 3, 2018, 11:13:29 PM)
url: http://34.218.227.254:8080/homework2/webapi/hw2server
Client starting.....Time: 2018/11/03 23:13:30
Warmup phase: All threads running.....
Warmup phase complete: Time 46.629 seconds
Loading phase: All threads running.....
Loading phase complete: Time 73.856 seconds
Peak phase: All threads running.....
Peak phase complete: Time 188.513 seconds
Cooldown phase: All threads running.....
Cooldown phase complete: Time 70.967 seconds
Client Execution Complete.

=====================Execution Summary=========================
Generating XYLineChart for the overall throughput...
XYLineChart WDDMThroughput.jpeg generated successfully.
Test Wall Time: 379.975 seconds
Total number of request sent: 240500
Total number of Successful responses: 240500
Overall throughput across all phases: 632.936 requests per second
Mean latencies for all requests: 32 MilliSeconds*
Median latencies for all requests: 28 MilliSeconds*
95th percentile latency for all requests: 48 MilliSeconds*
99th percentile latency for all requests: 131 MilliSeconds*
```

*client setting: **MaxThreads: 32**; Day number:1; User population: 100,000; Number of tests per phase: 100.*

XY-LineChart of overall throughput:

Note: The thread number in the above chart is the total number of threads generated during all four phases (e.g: 3 + 16 + 32 + 8 = 59).

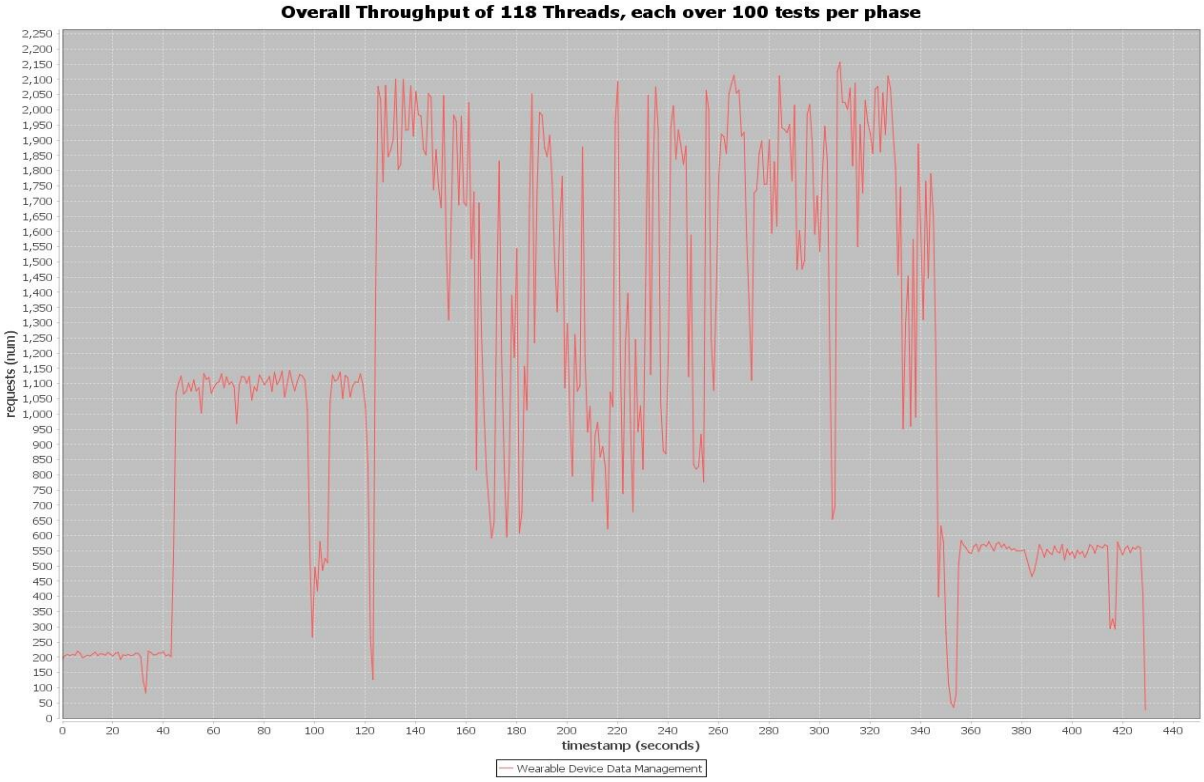
2.Performance Statistics: maxThread = 64

```
Markers  Properties  Servers  Data Source Explorer  Snippets  Console  Search
<terminated> WDDMClient [Java Application] C:\Program Files\Java\jre1.8.0_181\bin\javaw.exe (Nov 3, 2018, 11:21:47 PM)
url: http://34.218.227.254:8080/homework2/webapi/hw2server
Client starting.....Time: 2018/11/03 23:21:47
Warmup phase: All threads running.....
Warmup phase complete: Time 44.297 seconds
Loading phase: All threads running.....
Loading phase complete: Time 79.485 seconds
Peak phase: All threads running.....
Peak phase complete: Time 230.935 seconds
Cooldown phase: All threads running.....
Cooldown phase complete: Time 74.695 seconds
Client Execution Complete.

=====================Execution Summary=========================
Generating XYLineChart for the overall throughput...
XYLineChart WDDMThroughput.jpeg generated successfully.
Test Wall Time: 429.444 seconds
Total number of request sent: 481000
Total number of Successful responses: 481000
Overall throughput across all phases: 1,120.053 requests per second
Mean latencies for all requests: 37 MilliSeconds*
Median latencies for all requests: 31 MilliSeconds*
95th percentile latency for all requests: 62 MilliSeconds*
99th percentile latency for all requests: 312 MilliSeconds*
```

*client setting: **MaxThreads: 64**; Day number:1; User population: 100,000; Number of tests per phase: 100.*

XY-LineChart of overall throughput:



Note: The thread number in the above chart is the total number of threads generated during all four phases (e.g: 6 + 32 + 64 + 16 = 118).

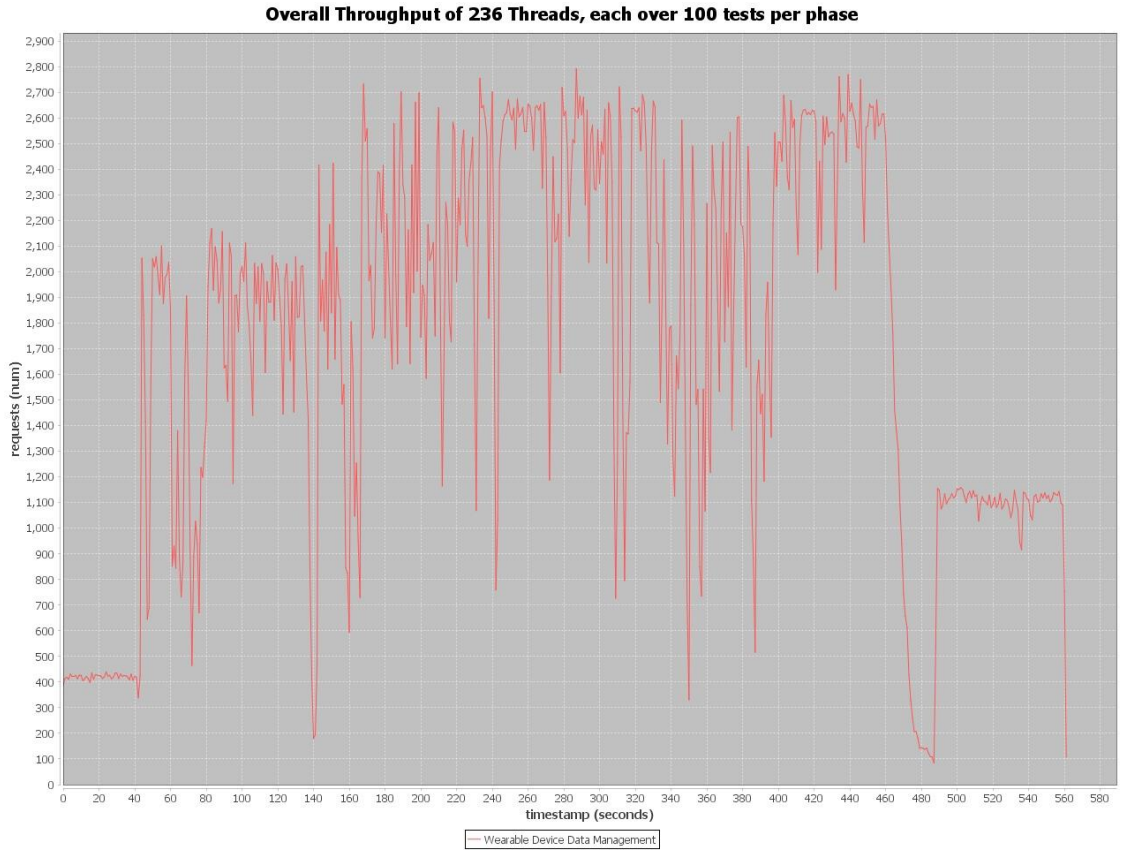
3.Performance Statistics: maxThread = 128

```
Markers  Properties  Servers  Data Source Explorer  Snippets  Console  Search
<terminated> WDDMClient [Java Application] C:\Program Files\Java\jre1.8.0_181\bin\javaw.exe (Nov 3, 2018, 11:31:12 PM)
url: http://34.218.227.254:8080/homework2/webapi/hw2server
Client starting.....Time: 2018/11/03 23:31:13
Warmup phase: All threads running.....
Warmup phase complete: Time 43.308 seconds
Loading phase: All threads running.....
Loading phase complete: Time 98.335 seconds
Peak phase: All threads running.....
Peak phase complete: Time 346.615 seconds
Cooldown phase: All threads running.....
Cooldown phase complete: Time 73.523 seconds
Client Execution Complete.

======================Execution Summary=========================
Generating XYLineChart for the overall throughput...
XYLineChart WDDMThroughput.jpeg generated successfully.
Test Wall Time: 561.799 seconds
Total number of request sent: 962000
Total number of Successful responses: 962000
Overall throughput across all phases: 1,712.356 requests per second
Mean latencies for all requests: 52 MilliSeconds*
Median latencies for all requests: 31 MilliSeconds*
95th percentile latency for all requests: 71 MilliSeconds*
99th percentile latency for all requests: 565 MilliSeconds*
```

*client setting: **MaxThreads: 128**; Day number:1; User population: 100,000; Number of tests per phase: 100.*

XY-LineChart of overall throughput:



Note: The thread number in the above chart is the total number of threads generated during all four phases (e.g: 12 + 64 +128 + 32 = 236)

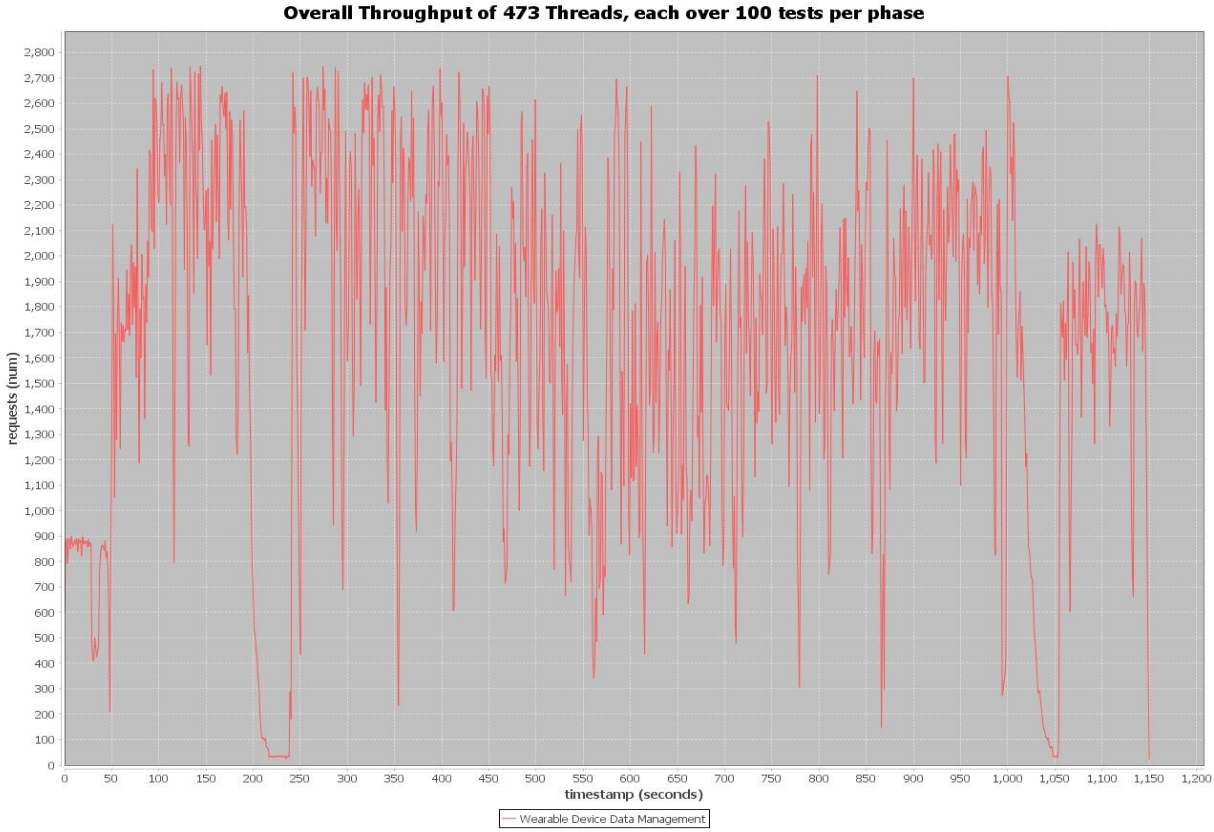
4.Performance Statistics: maxThread = 256

```
Markers  Properties  Servers  Data Source Explorer  Snippets  Console  Search
<terminated> WDDMClient [Java Application] C:\Program Files\Java\jre1.8.0_181\bin\javaw.exe (Nov 3, 2018, 11:48:06 PM)
url: http://34.218.227.254:8080/homework2/webapi/hw2server
Client starting.....Time: 2018/11/03 23:48:06
Warmup phase: All threads running.....
Warmup phase complete: Time 48.624 seconds
Loading phase: All threads running.....
Loading phase complete: Time 191.102 seconds
Peak phase: All threads running.....
Peak phase complete: Time 815.089 seconds
Cooldown phase: All threads running.....
Cooldown phase complete: Time 95.457 seconds
Client Execution Complete.

======================Execution Summary=========================
Generating XYLineChart for the overall throughput...
XYLineChart WDDMThroughput.jpeg generated successfully.
Test Wall Time: 1150.293 seconds
Total number of request sent: 1925500
Total number of Successful responses: 1925500
Overall throughput across all phases: 1,673.921 requests per second
Mean latencies for all requests: 112 MilliSeconds*
Median latencies for all requests: 31 MilliSeconds*
95th percentile latency for all requests: 331 MilliSeconds*
99th percentile latency for all requests: 1641 MilliSeconds*
```

*client setting: **MaxThreads: 256**; Day number:1; User population: 100,000; Number of tests per phase: 100.*

XY-LineChart of overall throughput:



Note: The thread number in the above chart is the total number of threads generated during all four phases (e.g: 25 + 128 + 256 + 64 = 473).

Step4: The plot and performance statistics of four different maxThreads values against Load Balancer:

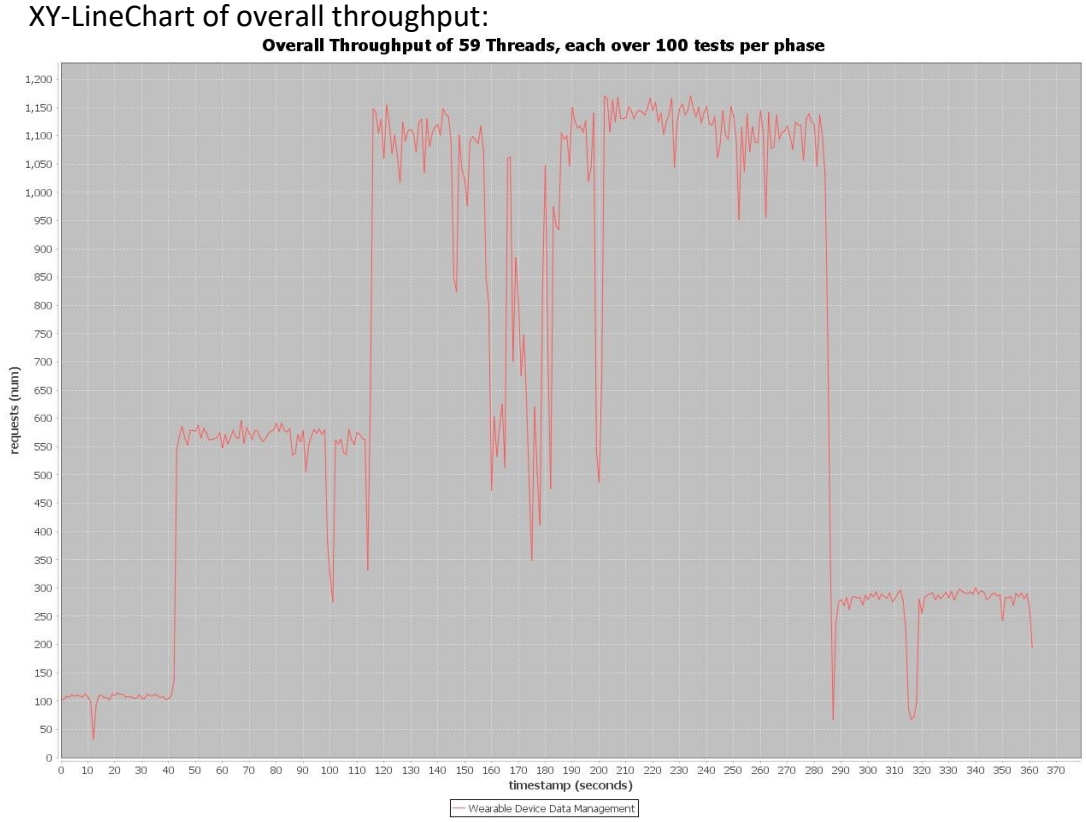
1. Performance Statistics: maxThread = 32

```
Markers  Properties  Servers  Data Source Explorer  Snippets  Console  Search
<terminated> WDDMClient [Java Application] C:\Program Files\Java\jre1.8.0_181\bin\javaw.exe (Nov 4, 2018, 12:11:13 AM)
url: http://WenWDDMLB-44716421.us-west-2.elb.amazonaws.com:8080/homework2/webapi/hw2server
Client starting.....Time: 2018/11/04 00:11:14
Warmup phase: All threads running.....
Warmup phase complete: Time 42.695 seconds
Loading phase: All threads running.....
Loading phase complete: Time 72.389 seconds
Peak phase: All threads running.....
Peak phase complete: Time 173.061 seconds
Cooldown phase: All threads running.....
Cooldown phase complete: Time 73.753 seconds
Client Execution Complete.

=====================Execution Summary=========================
Generating XYLineChart for the overall throughput...
XYLineChart WDDMThroughput.jpeg generated successfully.
Test Wall Time: 361.924 seconds
Total number of request sent: 240500
Total number of Successful responses: 240500
Overall throughput across all phases: 664.504 requests per second
Mean latencies for all requests: 30 MilliSeconds*
Median latencies for all requests: 31 MilliSeconds*
95th percentile latency for all requests: 47 MilliSeconds*
99th percentile latency for all requests: 98 MilliSeconds*
```

*client setting: **MaxThreads: 32**; Day number:1; User population: 100,000; Number of tests per phase: 100.*

XY-LineChart of overall throughput:



Note: The thread number in the above chart is the total number of threads generated during all four phases (e.g: 3 + 16 + 32 + 8 = 59).

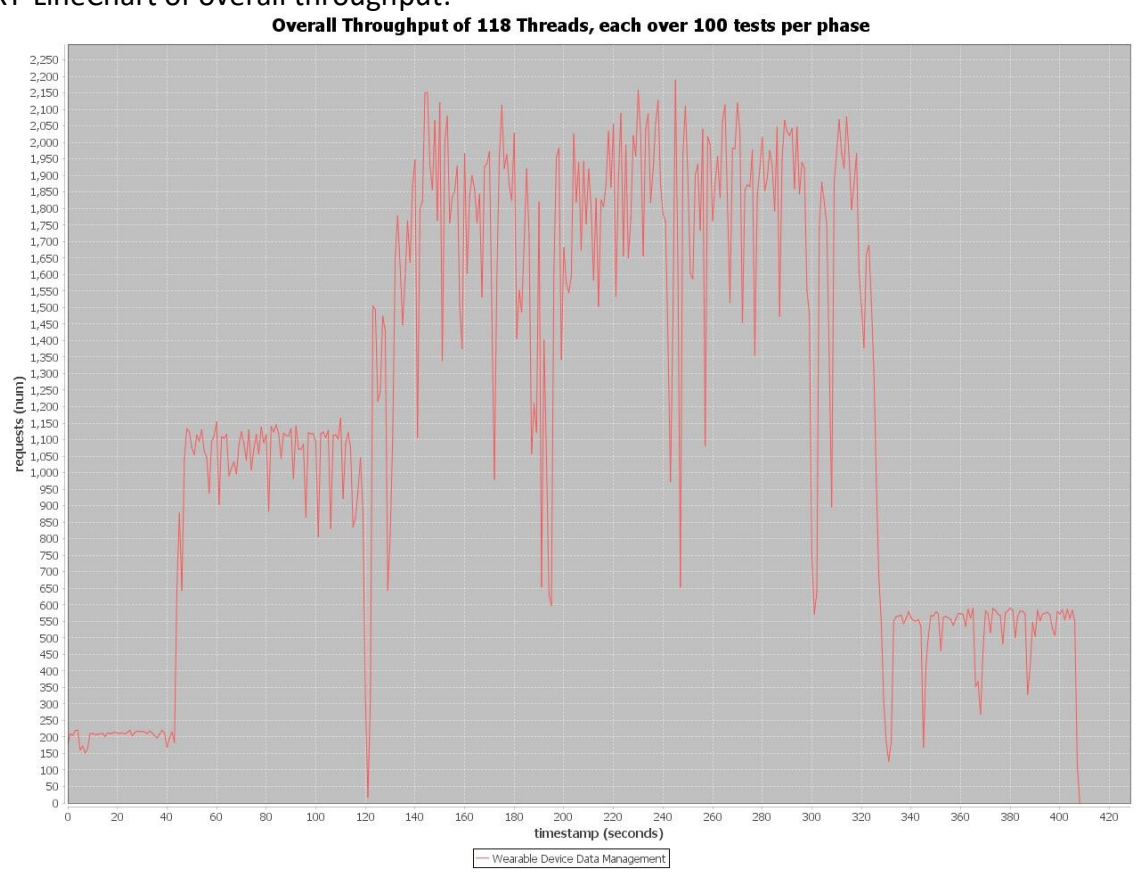
2.Performance Statistics: maxThread = 64

```
Markers  Properties  Servers  Data Source Explorer  Snippets  Console  Search
<terminated> WDDMClient [Java Application] C:\Program Files\Java\jre1.8.0_181\bin\javaw.exe (Nov 4, 2018, 12:49:52 PM)
url: http://WenWDDMLB-44716421.us-west-2.elb.amazonaws.com:8080/homework2/webapi/hw2server
Client starting.....Time: 2018/11/04 12:49:52
Warmup phase: All threads running.....
Warmup phase complete: Time 44.068 seconds
Loading phase: All threads running.....
Loading phase complete: Time 77.958 seconds
Peak phase: All threads running.....
Peak phase complete: Time 210.598 seconds
Cooldown phase: All threads running.....
Cooldown phase complete: Time 75.413 seconds
Client Execution Complete.

======================Execution Summary==========================
Generating XYLineChart for the overall throughput...
XYLineChart WDDMThroughput.jpeg generated successfully.
Test Wall Time: 408.049 seconds
Total number of request sent: 481000
Total number of Successful responses: 481000
Overall throughput across all phases: 1,178.78 requests per second
Mean latencies for all requests: 35 MilliSeconds*
Median latencies for all requests: 28 MilliSeconds*
95th percentile latency for all requests: 57 MilliSeconds*
99th percentile latency for all requests: 280 MilliSeconds*
```

*client setting: **MaxThreads: 64**; Day number:1; User population: 100,000; Number of tests per phase: 100.*

XY-LineChart of overall throughput:



Note: The thread number in the above chart is the total number of threads generated during all four phases (e.g: 6 + 32 + 64 + 16 = 118).

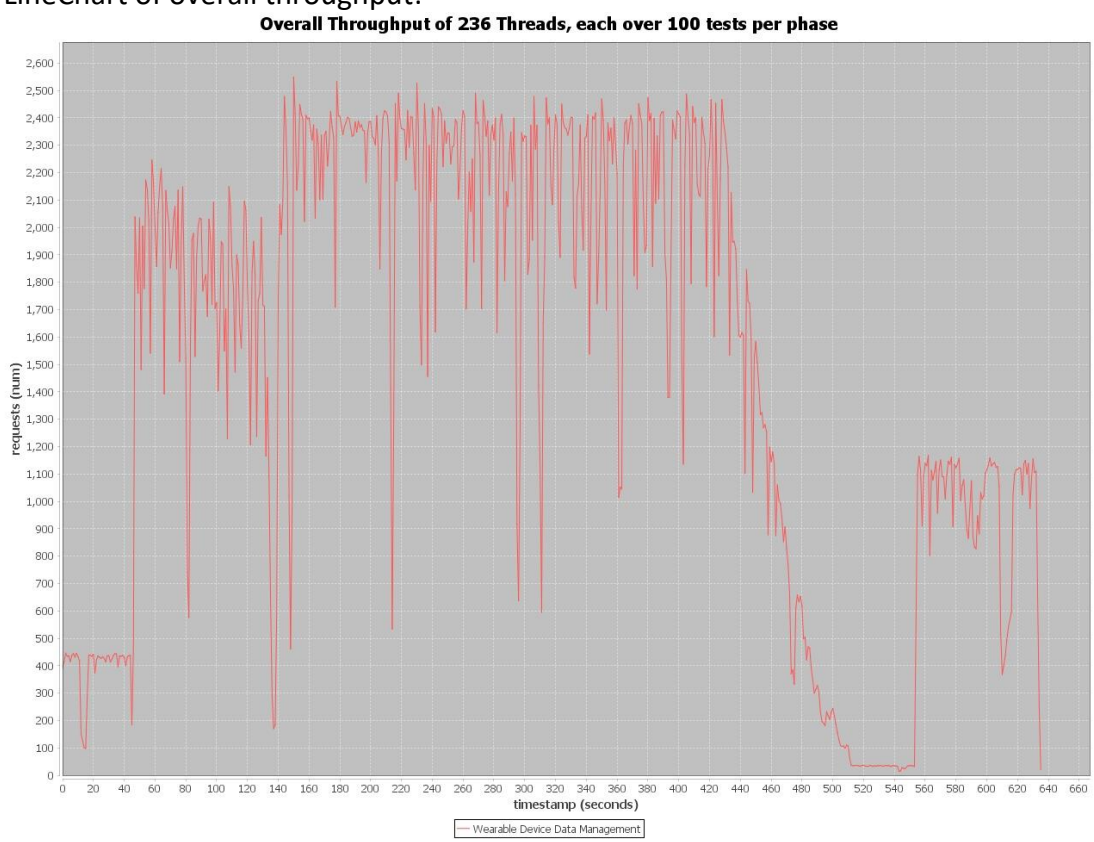
3.Performance Statistics: maxThread = 128

```
Markers  Properties  Servers  Data Source Explorer  Snippets  Console  Search
<terminated> WDDMClient [Java Application] C:\Program Files\Java\jre1.8.0_181\bin\javaw.exe (Nov 4, 2018, 12:58:38 PM)
url: http://WenWDDMLB-44716421.us-west-2.elb.amazonaws.com:8080/homework2/webapi/hw2server
Client starting.....Time: 2018/11/04 12:58:39
Warmup phase: All threads running.....
Warmup phase complete: Time 45.906 seconds
Loading phase: All threads running.....
Loading phase complete: Time 92.831 seconds
Peak phase: All threads running.....
Peak phase complete: Time 415.309 seconds
Cooldown phase: All threads running.....
Cooldown phase complete: Time 81.335 seconds
Client Execution Complete.

=====================Execution Summary=========================
Generating XYLineChart for the overall throughput...
XYLineChart WDDMThroughput.jpeg generated successfully.
Test Wall Time: 635.411 seconds
Total number of request sent: 962000
Total number of Successful responses: 962000
Overall throughput across all phases: 1,513.981 requests per second
Mean latencies for all requests: 50 MilliSeconds*
Median latencies for all requests: 30 MilliSeconds*
95th percentile latency for all requests: 69 MilliSeconds*
99th percentile latency for all requests: 487 MilliSeconds*
```

*client setting: **MaxThreads: 128**; Day number:1; User population: 100,000; Number of tests per phase: 100.*

XY-LineChart of overall throughput:



Note: The thread number in the above chart is the total number of threads generated during all four phases (e.g: 12 + 64 +128 + 32 = 236)

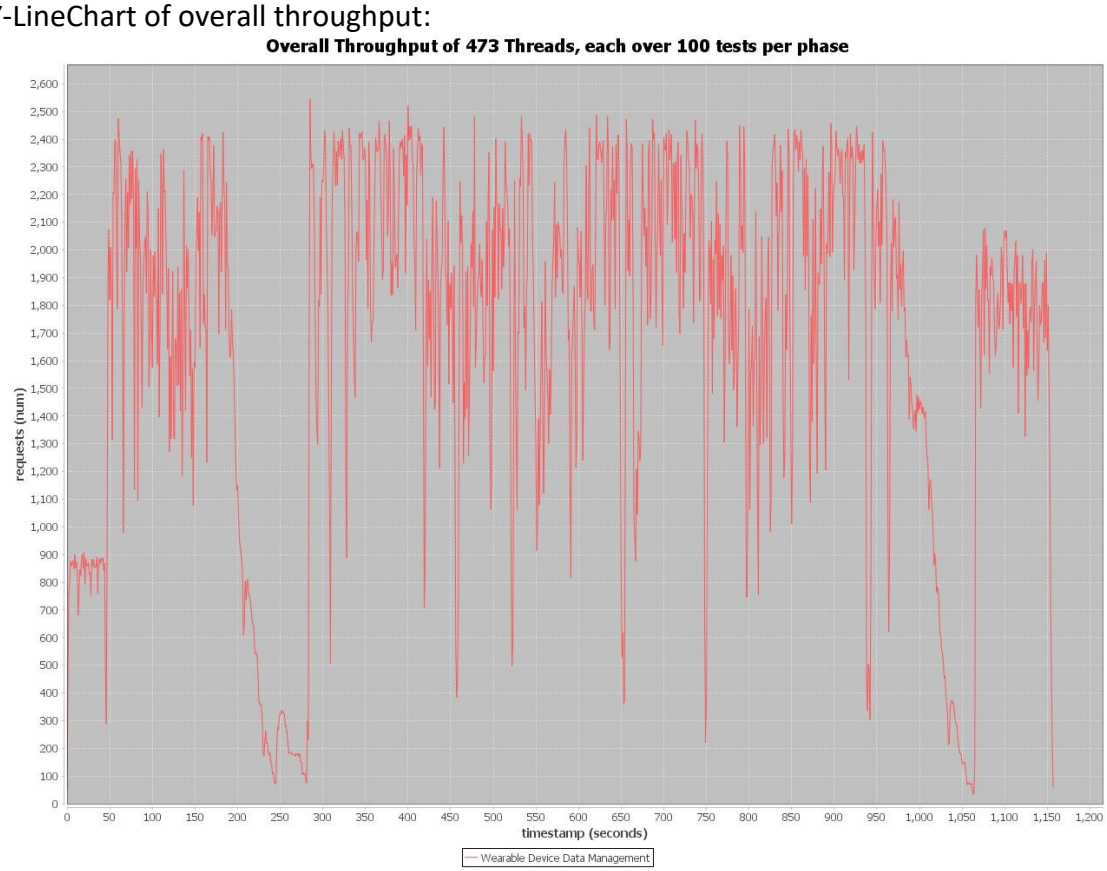
4.Performance Statistics: maxThread = 256

```
Markers  Properties  Servers  Data Source Explorer  Snippets  Console  Search
<terminated> WDDMClient [Java Application] C:\Program Files\Java\jre1.8.0_181\bin\javaw.exe (Nov 3, 2018, 3:22:32 PM)
url: http://WenWDDMLB-44716421.us-west-2.elb.amazonaws.com:8080/homework2/webapi/hw2server
Client starting.....Time: 2018/11/03 15:22:33
Warmup phase: All threads running.....
Warmup phase complete: Time 46.228 seconds
Loading phase: All threads running.....
Loading phase complete: Time 236.561 seconds
Peak phase: All threads running.....
Peak phase complete: Time 782.698 seconds
Cooldown phase: All threads running.....
Cooldown phase complete: Time 92.416 seconds
Client Execution Complete.

======================Execution Summary=========================
Generating XYLineChart for the overall throughput...
XYLineChart WDDMThroughput.jpeg generated successfully.
Test Wall Time: 1157.92 seconds
Total number of request sent: 1925500
Total number of Successful responses: 1925500
Overall throughput across all phases: 1,662.896 requests per second
Mean latencies for all requests: 101 MilliSeconds*
Median latencies for all requests: 31 MilliSeconds*
95th percentile latency for all requests: 174 MilliSeconds*
99th percentile latency for all requests: 974 MilliSeconds*
```

*client setting: **MaxThreads: 256**; Day number:1; User population: 100,000; Number of tests per phase: 100.*

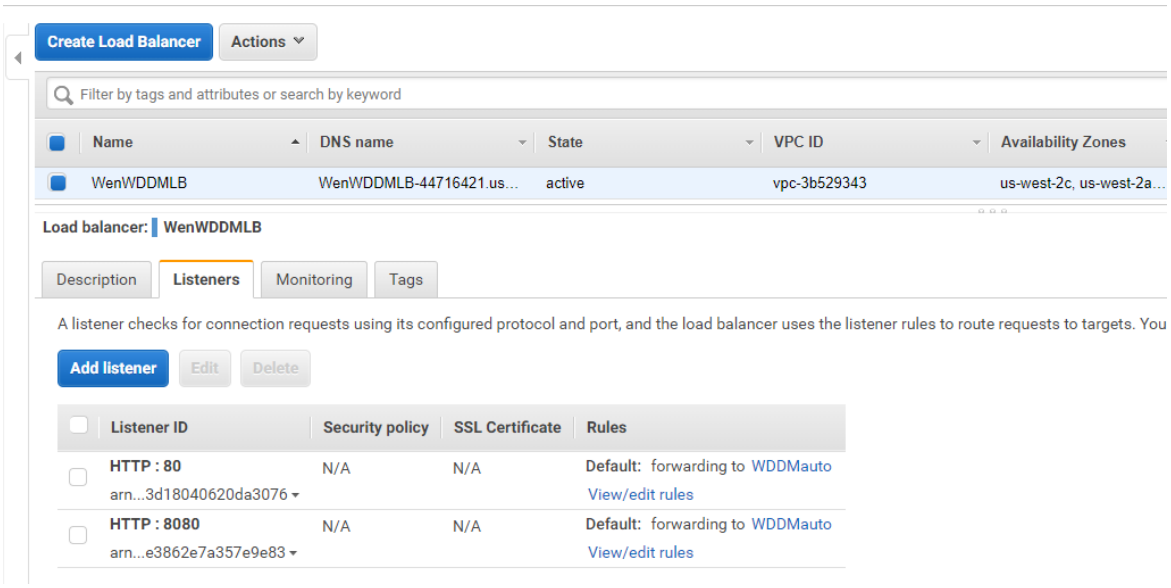
XY-LineChart of overall throughput:



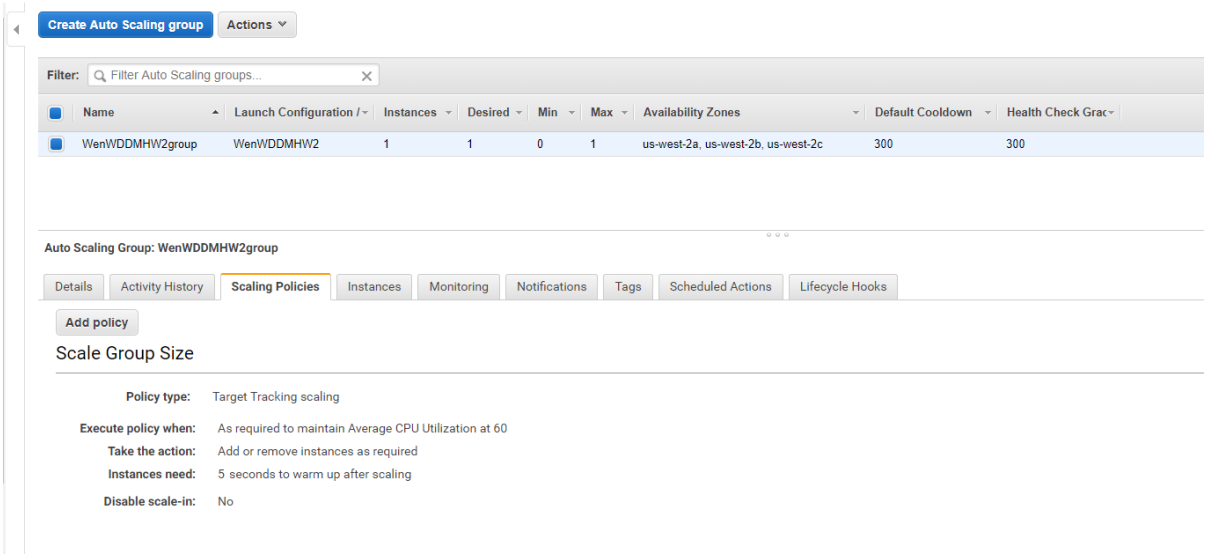
Note: The thread number in the above chart is the total number of threads generated during all four phases (e.g: 25 + 128 + 256 + 64 = 473).

My Load Balancer Setup:

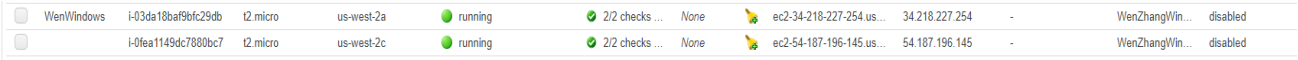
I setup my below AWS load balancer instance and let it listens to HTTP requests from port 8080 and 80. I also created an autoscaling group and linked it to the load balancer. The autoscaling group has a scaling policy that whenever the average CPU utilization of the group instances raise above 60% for more than 5 seconds, a new EC2 instance will be created to load balance traffic. I created the AMI image from my original EC2 instance so that it can be used to create the new EC2 instance when the autoscaling policy is been triggered. Below are screenshots of my load balancer instance, autoscaling group and its policy, and my EC2 instances (one original, one created by autoscaling when the policy is been hit).



Load Balancer Instance



Auto Scaling group that linked with the load balancer



My original EC2 Instance and auto generated duplicate instance

Performance comparison between Step3 and Step4:

When running maxThreads number of 128 and 256, the CPU Utilization of my original EC2 instance increased up to 69% and 75% which resulted in triggering the autoscaling rule. A new EC2 instance was created based on the AMI of the original EC2 instance. However, I met a small problem during this process that when AMI was been created, the Tomcat server on the original EC2 instance was forced to stop. Thus, when the new instance was been created by the autoscaling policy, it wasn't able to immediately start handling the traffic. I had to manually connect to the new instance and start its Tomcat service in order to let it load balancing. This makes that certain run incomparable so I re-ran the test. For step4 comparison, I then also ran the settings of maxThreads values of 32 and 64 against load balancer and generated the results in previous pages.

After comparing the test results between step3 (single EC2 instance) and step4 (Load Balancer with 2 EC2 instances) for my case, I found out that the performance with single EC2 instance and with load balancer are actually almost the same. I think this is because the benefit of utilizing the second EC2 instance resource to gain higher throughput almost matches the extra latencies introduced by the load balancer during decision making and data forwarding. Also, since even during 128 and 256 maxThreads cases, the maximum CPU utilization of my single EC2 instance was 75%, which means the instance was still capable of handling more requests before throughput become worse.

Step5: Bonus, Scaling up

I first tried 512 maxThreads, 100 iterations setting against load balancer. However, during the test, I saw a lot of time out exceptions. Below is the running result:

```
Markers  Properties  Servers  Data Source Explorer  Snippets  Console  Search
<terminated> WDDMClient [Java Application] C:\Program Files\Java\jre1.8.0_181\bin\javaw.exe (Nov 4, 2018, 1:36:13 PM)
url: http://WenWDDMLB-44716421.us-west-2.elb.amazonaws.com:8080/homework2/webapi/hw2server
Client starting.....Time: 2018/11/04 13:36:14
Warmup phase: All threads running.....
Warmup phase complete: Time 54.212 seconds
Loading phase: All threads running.....
javax.ws.rs.ProcessingException: java.net.SocketTimeoutException: Read timed out
        at org.glassfish.jersey.client.internal.HttpUrlConnector.apply(HttpUrlConnector.java:284)
        at org.glassfish.jersey.client.ClientRuntime.invoke(ClientRuntime.java:278)
        at org.glassfish.jersey.client.JerseyInvocation.lambda$invoke$0(JerseyInvocation.java:753)
        at org.glassfish.jersey.internal.Errors.process(Errors.java:316)
        at org.glassfish.jersey.internal.Errors.process(Errors.java:298)
        at org.glassfish.jersey.internal.Errors.process(Errors.java:229)
        at org.glassfish.jersey.process.internal.RequestScope.runInScope(RequestScope.java:414)
        at org.glassfish.jersey.client.JerseyInvocation.invoke(JerseyInvocation.java:752)
        at org.glassfish.jersey.client.JerseyInvocation$Builder.method(JerseyInvocation.java:445)
        at org.glassfish.jersey.client.JerseyInvocation$Builder.post(JerseyInvocation.java:351)
        at wenzhang.cs6650.homework2.client.Worker.postIt(Worker.java:203)
        at wenzhang.cs6650.homework2.client.Worker.run(Worker.java:97)
        at java.lang.Thread.run(Unknown Source)
Caused by: java.net.SocketTimeoutException: Read timed out
        at java.net.SocketInputStream.socketRead0(Native Method)
        at java.net.SocketInputStream.socketRead(Unknown Source)
        at java.net.SocketInputStream.read(Unknown Source)
        at java.net.SocketInputStream.read(Unknown Source)
        at java.io.BufferedInputStream.fill(Unknown Source)
        at java.io.BufferedInputStream.read1(Unknown Source)
        at java.io.BufferedInputStream.read(Unknown Source)
        at sun.net.www.http.HttpClient.parseHTTPHeader(Unknown Source)
        at sun.net.www.http.HttpClient.parseHTTP(Unknown Source)
        at sun.net.www.protocol.http.HttpURLConnection.getInputStream0(Unknown Source)
        at sun.net.www.protocol.http.HttpURLConnection.getInputStream(Unknown Source)
        at java.net.HttpURLConnection.getResponseCode(Unknown Source)
        at org.glassfish.jersey.client.internal.HttpUrlConnector._apply(HttpUrlConnector.java:390)
        at org.glassfish.jersey.client.internal.HttpUrlConnector.apply(HttpUrlConnector.java:282)
        ... 12 more
Loading phase complete: Time 395.025 seconds
Peak phase: All threads running.....
javax.ws.rs.ProcessingException: java.net.ConnectException: Connection timed out: connect
        at org.glassfish.jersey.client.internal.HttpUrlConnector.apply(HttpUrlConnector.java:284)
        at org.glassfish.jersey.client.ClientRuntime.invoke(ClientRuntime.java:278)
        at org.glassfish.jersey.client.JerseyInvocation.lambda$invoke$0(JerseyInvocation.java:753)
        at org.glassfish.jersey.internal.Errors.process(Errors.java:316)
        at org.glassfish.jersey.internal.Errors.process(Errors.java:298)
        at org.glassfish.jersey.internal.Errors.process(Errors.java:229)
        at org.glassfish.jersey.process.internal.RequestScope.runInScope(RequestScope.java:414)
        at org.glassfish.jersey.client.JerseyInvocation.invoke(JerseyInvocation.java:752)
        at org.glassfish.jersey.client.JerseyInvocation$Builder.method(JerseyInvocation.java:445)
        at org.glassfish.jersey.client.JerseyInvocation$Builder.post(JerseyInvocation.java:351)
        at wenzhang.cs6650.homework2.client.Worker.postIt(Worker.java:203)
        at wenzhang.cs6650.homework2.client.Worker.run(Worker.java:86)
        at java.lang.Thread.run(Unknown Source)
Caused by: java.net.ConnectException: Connection timed out: connect
```

When running 512 maxThreads, I saw a lot of connection time out at Socket level. This means that my local machine cannot hold 512 maxThreads simultaneously, so the bottleneck of the system is my local machine. Thus, I chose to increase number of iterations instead of maxThreads for scale up testing.

During the running of all the tests in step3 and step4, I captured their corresponding CPU utilization. The results are below:

Single EC2

Threads	CPU %	DB %
32	28%	34%
64	43%	46%
128	75%	53%
256	69%	54%

Load Balancer

Threads	CPU %	DB %
32	11%, 10%	35%
64	20%, 26%	44%
128	35%, 38%	51%
256	35%, 37%	49%

From the above two tables, I see that when the maxThread value is 128, the CPU utilization of EC2 instance has the largest number of 75%. Thus, to test the worst-case scenario, I tested the scale up with the setting of maxThreads = 128, and number of iterations = 1000. Below is the test result:

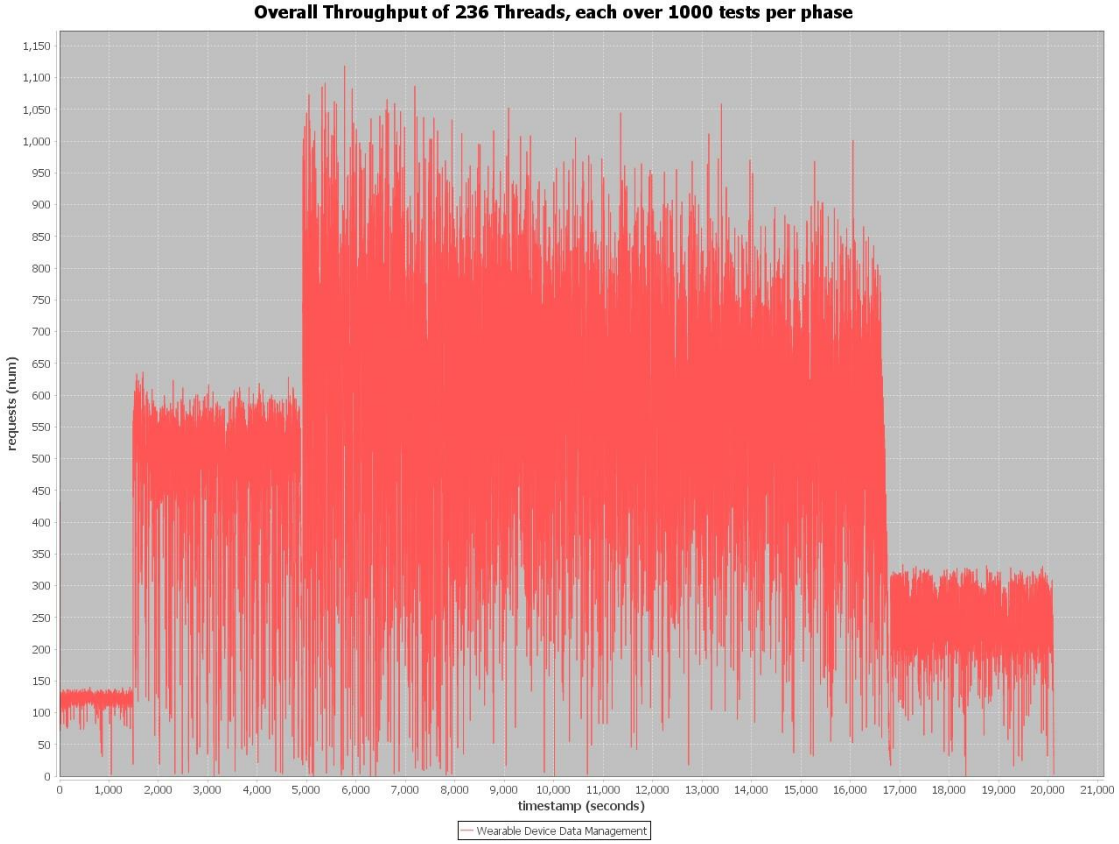
Performance Statistics: maxThread = 128

```
Markers  Properties  Servers  Data Source Explorer  Snippets  Console  Search
<terminated> WDDMClient [Java Application] C:\Program Files\Java\jre1.8.0_181\bin\javaw.exe (Nov 4, 2018, 3:33:13 PM)
url: http://WenWDDMLB-44716421.us-west-2.elb.amazonaws.com:8080/homework2/webapi/hw2server
Client starting.....Time: 2018/11/04 15:33:13
Warmup phase: All threads running.....
Warmup phase complete: Time 1487.147 seconds
Loading phase: All threads running.....
Loading phase complete: Time 3433.778 seconds
Peak phase: All threads running.....
Peak phase complete: Time 11892.236 seconds
Cooldown phase: All threads running.....
Cooldown phase complete: Time 3297.982 seconds
Client Execution Complete.

=====================Execution Summary=========================
Generating XYLineChart for the overall throughput...
XYLineChart WDDMThroughput.jpeg generated successfully.
Test Wall Time: 20111.191 seconds
Total number of request sent: 9620000
Total number of Successful responses: 9620000
Overall throughput across all phases: 478.341 requests per second
Mean latencies for all requests: 192 MilliSeconds*
Median latencies for all requests: 171 MilliSeconds*
95th percentile latency for all requests: 594 MilliSeconds*
99th percentile latency for all requests: 1082 MilliSeconds*
```

client setting: ***MaxThreads: 128****; Day number:1; User population: 100,000;* ***Number of tests per phase: 1000****.*

XY-LineChart of overall throughput:



Note: The thread number in the above chart is the total number of threads generated during all four phases (e.g: 12 + 64 +128 + 32 = 236)