COMPX341-20A

Assignment 5 - Dominic Maas - 1344405

# Testing Environment

The following hardware was used to run the following tests. Originally the test was run on a different PC (using docker with the WSL2 backend), but there were socket connection issues while running JMeter. Instead tests were run from a Windows 10 PC across a local network (via gigabyte wired connection) to a separate server (running Ubuntu Server).

## Server

* **CPU**: AMD Ryzen 5 3400G (8) @ 3.7GHz
* **Memory**: 32GB DDR4
* **Operating System:** Ubuntu 20.04 LTS x86\_64
* **Kernel:** 5.4.0-37-generic
* **Docker:** 19.03.8, build afacb8b7f0

## Client

* **JMeter:** 5.3
* **Java:** 1.8.0

# Testing Methodology

Before running any tests, ensure “docker-compose down” is run first to generate a blank testing slate (otherwise results from previous tests may impact the currently running test).

## Scenario 1

1. Ensure all limits are removed from the “docker-compose.yml” file
2. Run the following commands: “docker-compose down”, “docker-compose build”, “docker-compose up”. This will ensure a clean slate with the latest changes
3. Open “scenario\_1.jmx” within JMeter (5.3 was used for the following results)
4. Click on “Scenario 1” then click on the green start button to begin performance testing
5. Repeat steps 2-4 adding a CPU limit of 0.5 to the “docker-compose.yml” file
6. Repeat steps 2-4 adding a CPU limit of 0.01 to the “docker-compose.yml” file

Once complete, you should have 3 sets of data, one for no CPU limit, one for 0.5 CPU limit and a final dataset for 0.01 CPU limit. CPU limits are defined as follows: 0.5 -> 50% of a single core.

## Scenario 2

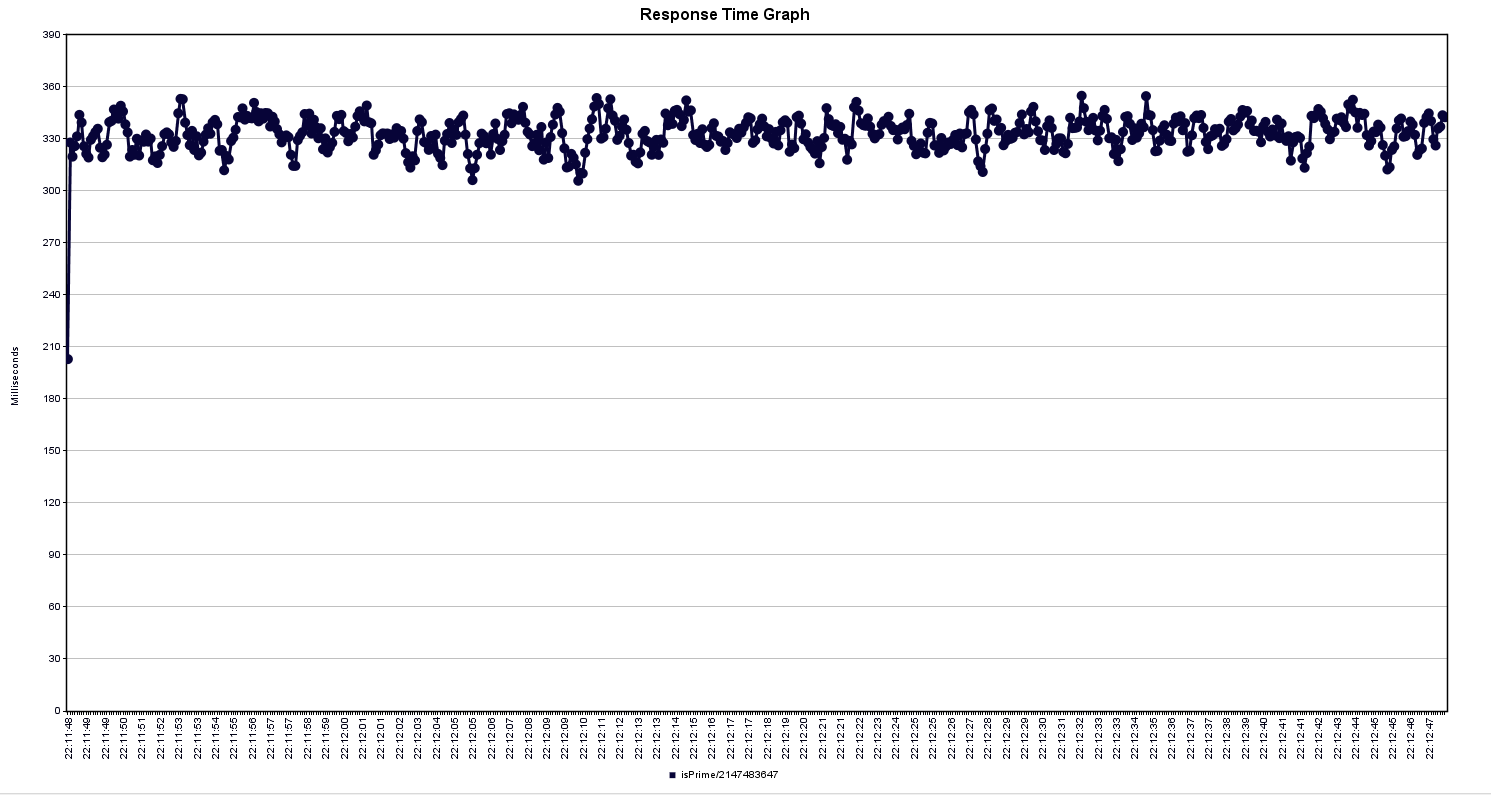
1. Ensure all limits are removed from the “docker-compose.yml” file
2. Run the following commands: “docker-compose down”, “docker-compose build”, “docker-compose up”. This will ensure a clean slate with the latest changes
3. Open “scenario\_2.jmx” within JMeter (5.3 was used for the following results)
4. Click on “Scenario 2” then click on the green start button to begin performance testing (the thread groups will be run in sequential order)
5. Repeat steps 2-4 adding a CPU limit of 0.5 to the “docker-compose.yml” file
6. Repeat steps 2-4 adding a CPU limit of 0.01 to the “docker-compose.yml” file

Once complete, you should have 6 sets of data, two for no CPU limit, two for 0.5 CPU limit and a final two datasets for 0.01 CPU limit. CPU limits are defined as follows: 0.5 -> 50% of a single core.

# Testing Scenario 1

## No Limits

* **Throughput:** 150.1 / second
* **Response Time:**
  + **Average:** 332ms
  + **Min:** 39ms
  + **Max:** 337ms



# Testing Scenario 2

…

aTo fix http:

bin.user.ptoperties

httpclient4.retrycount=1

hc.parameters.file=hc.parameters

h.parametesr

http.connection.stalecheck$Boolean=true