CCBD

DOCKER PERFORMANCE EVALUATION

What is docker?

- Docker is a tool designed to make it easier to create, deploy, and run applications by using containers.
- Containers allow a developer to package up an application with all of the parts it needs, such as libraries and other dependencies, and deploy it as one package.
- **Purpose of Docker:** Its primary focus is to automate the deployment of applications inside software containers and the automation of operating system level virtualization on Linux. It's more lightweight than standard Containers and boots up in seconds.

Project Objective

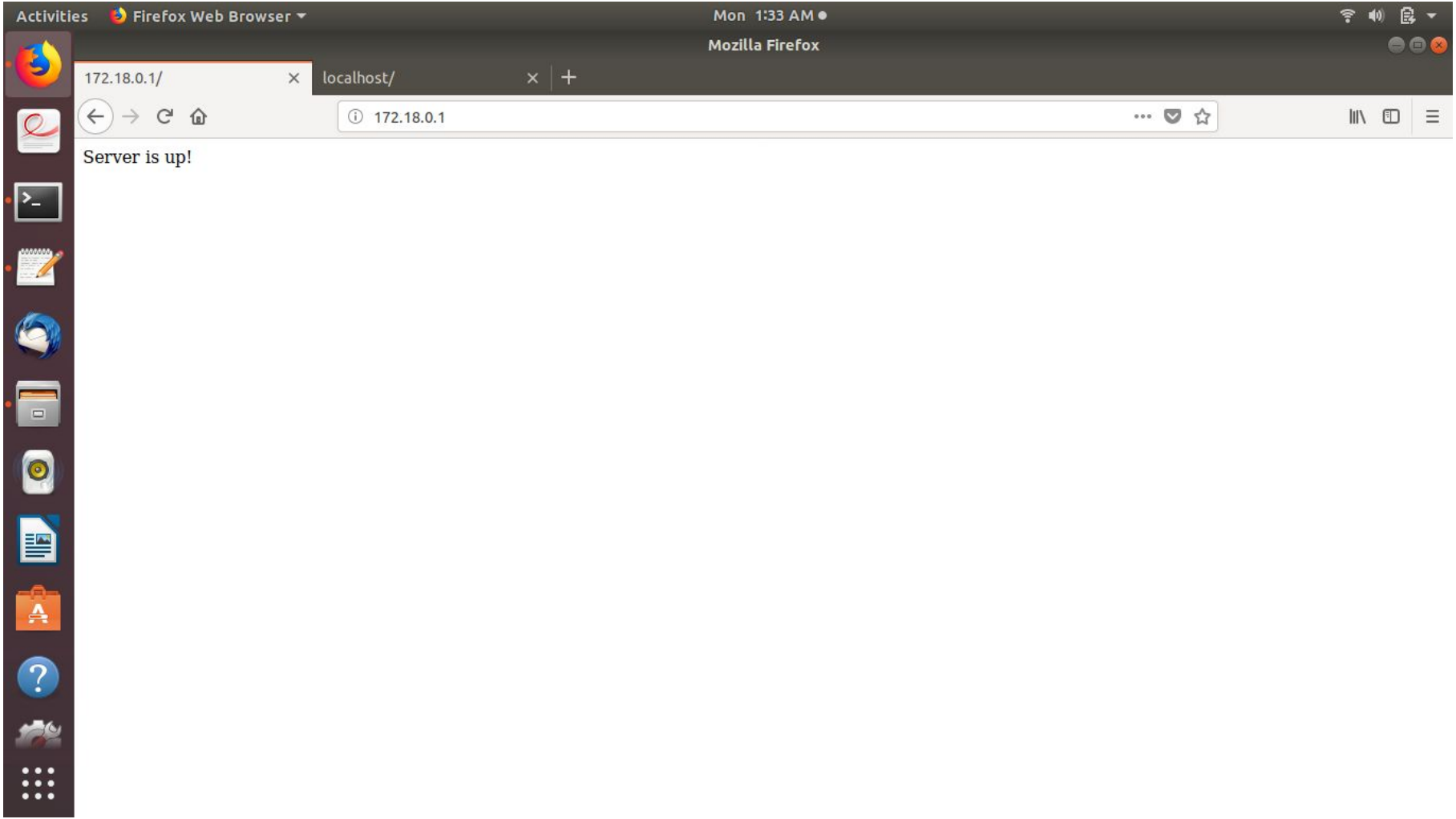
- Create two docker containers each for server and client.
- Send HTTP request from client to server.
- Compare the performance with and without docker using any performance testing tool.

Testing the Performance

Setting up the client and server:

```
user1@user1-Lenovo-G580:~$ sudo docker-compose build
Building server
Step 1/4 : FROM python:latest
 ---> 7f5b6ccd03e9
Step 2/4 : ADD server.py /server/
 ---> Using cache
 ---> 41a91a911128
Step 3/4 : ADD index.html /server/
 ---> Using cache
 ---> 7fab630757e4
Step 4/4 : WORKDIR /server/
 ---> Using cache
 ---> b8962d3ce1e9
Successfully built b8962d3ce1e9
Successfully tagged user1_server:latest
Building client
Step 1/3 : FROM python:latest
 ---> 7f5b6ccd03e9
Step 2/3 : ADD client.py /client/
 ---> Using cache
 ---> 8a3a240357ac
Step 3/3 : WORKDIR /client/
 ---> Using cache
 ---> 5b3aeb970079
Successfully built 5b3aeb970079
Successfully tagged user1_client:latest
user1@user1-Lenovo-G580:~$ sudo docker-compose up
Starting user1_server_1 ...
Starting user1_server_1 ... done
Starting user1_client_1 ...
Starting user1_client_1 ... done
Attaching to user1_server_1, user1_client_1
client_1  | Server is up!
client_1  |
```

With Docker


Activities
Firefox Web Browser
Mon 1:33 AM
Mozilla Firefox
172.18.0.1/
localhost/
172.18.0.1
Server is up!

Activities Terminal Mon 1:34 AM

user1@user1-Lenovo-G580: ~

File Edit View Search Terminal Tabs Help

user1@user1-Lenovo-G580: ~ | user1@user1-Lenovo-G580: ~

3 users, ~0 active

Latest Interval Stats at 20:04:23

Average Times:		Percentiles:		Response Codes:	
Elapsed:	0.002	0.0%:	0.001	200:	100.00% (175)
Connect:	0.000	50.0%:	0.002	All:	100.00% (175)
Latency:	0.002	90.0%:	0.002		
		95.0%:	0.003		
		99.0%:	0.004		
		99.9%:	0.005		
		100.0%:	0.005		

Taurus

\ v1.14.2 by BlazeMeter.com \

JMeter: linear-growth

35 %

Elapsed: 00:00:42 ETA: 00:01:17

175 hits, 0 fail

Cumulative Stats 00:00:35

Average Times:		Percentiles:		Response Codes:	
Elapsed:	0.002	0.0%:	0.001	200:	100.00% (3220)
Connect:	0.001	50.0%:	0.002	All:	100.00% (3220)
Latency:	0.002	90.0%:	0.003		
		95.0%:	0.003		
		99.0%:	0.005		
		99.9%:	0.026		
		100.0%:	1.042		

local

engine-loop:	0.079
bytes-sent:	99,813
cpu:	49.700
disk-read:	0
conn-all:	0
bytes-recv:	112,897
disk-space:	92.000
mem:	68.900
disk-write:	290,482

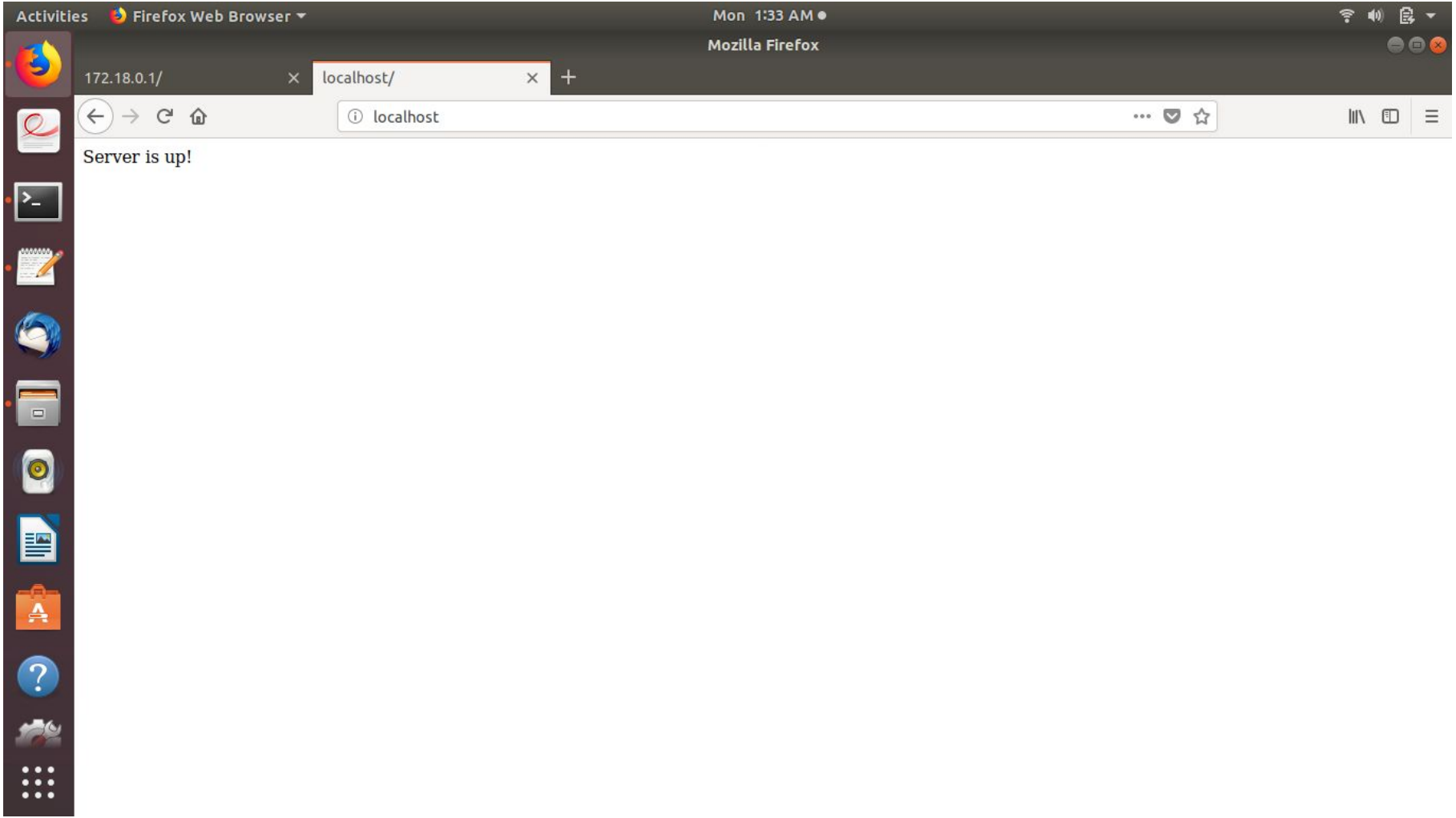
Labels	Hits	Failures	Avg Time
http://172.18.0.1/	3220	0.00%	0.002

Errors:

No failures occured

20:03:45 INFO: Waiting for finish...

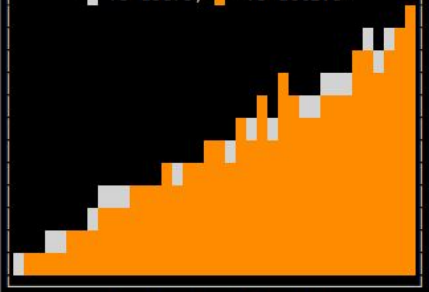
Without Docker


Activities
Firefox Web Browser
Mon 1:33 AM
Mozilla Firefox
172.18.0.1/
localhost/
localhost
Server is up!

Activities Tk Mon 1:26 AM

Taurus Status 161x41

75 users, ~75 active



Latest Interval Stats at 01:25:56

Average Times:		Percentiles:		Response Codes:	
Elapsed:	0.077	0.0%:	0.001	200:	100.00% (1183)
Connect:	0.000	50.0%:	0.005	All:	100.00% (1183)
Latency:	0.077	90.0%:	0.009		
		95.0%:	0.016		
		99.0%:	3.028		
		99.9%:	7.324		
		100.0%:	7.528		

Taurus

/ v1.14.2 by BlazeMeter.com /

JMeter: quick-test

18 %

Elapsed: 00:01:05 ETA: 00:04:54

local

disk-read:	0
bytes-recv:	2,340,532
cpu:	90.100
disk-space:	91.900
disk-write:	10,564,197
mem:	79.500
bytes-sent:	2,274,349
conn-all:	207
engine-loop:	3.033

1183 hits, 0 fail

Cumulative Stats 00:00:37

Average Times:		Percentiles:		Response Codes:	
Elapsed:	0.035	0.0%:	0.001	200:	100.00% (35761)
Connect:	0.001	50.0%:	0.005	All:	100.00% (35761)
Latency:	0.034	90.0%:	0.010		
		95.0%:	0.015		
		99.0%:	1.028		
		99.9%:	3.042		
		100.0%:	7.528		

0.077 avg time (lat, conn)

Labels	Hits	Failures	Avg Time
http://localhost/	35761	0.00%	0.035

Errors:

No failures occured

```
[2020-07-20 01:26:23,534 DEBUG Engine]
Checking
<bzt.modules.reporting.FinalStatus
object at 0x7f5bf66d8d30>
[2020-07-20 01:26:23,618 DEBUG Engine]
Checking <bzt.modules.console.ConsoleSta
tusReporter object at 0x7f5bf6754b00>
[2020-07-20 01:26:23,693 DEBUG
Engine.console] Current: 75 vu 1183
succ 0 fail 0.077 avg rt / Cumulative:
0.035 avg rt, 0% failures
```

Comparing the Performance

Test duration: 2 minutes

Tool used: Taurus

Sample count

With Docker: 26619

Without Docker: 102364

Failure

With Docker: 0%

Without Docker: 0.01

Inference:

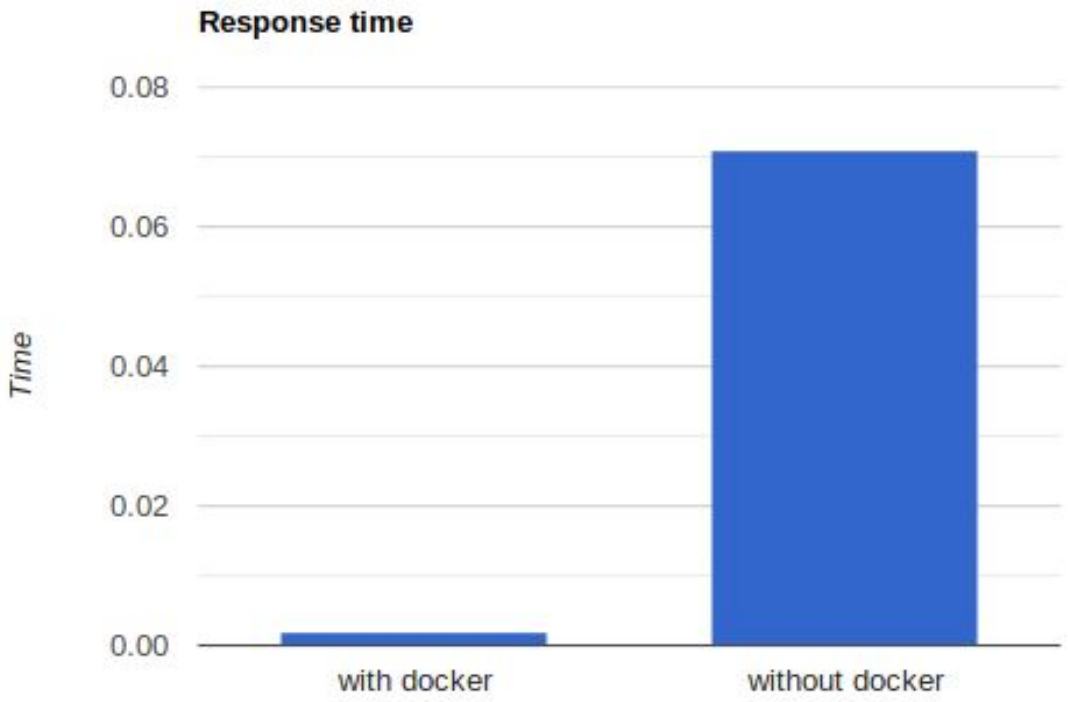
Failure chances is slightly more for without docker.

Average time

Average Response Time. **Response time** refers to the amount of **time** Application Server takes to return the results of a request to the user. The **response time** is affected by factors such as network bandwidth, number of users, number and type of requests submitted, and **average** think **time**.

With Docker: 0.002

Without Docker: 0.073



Inference:

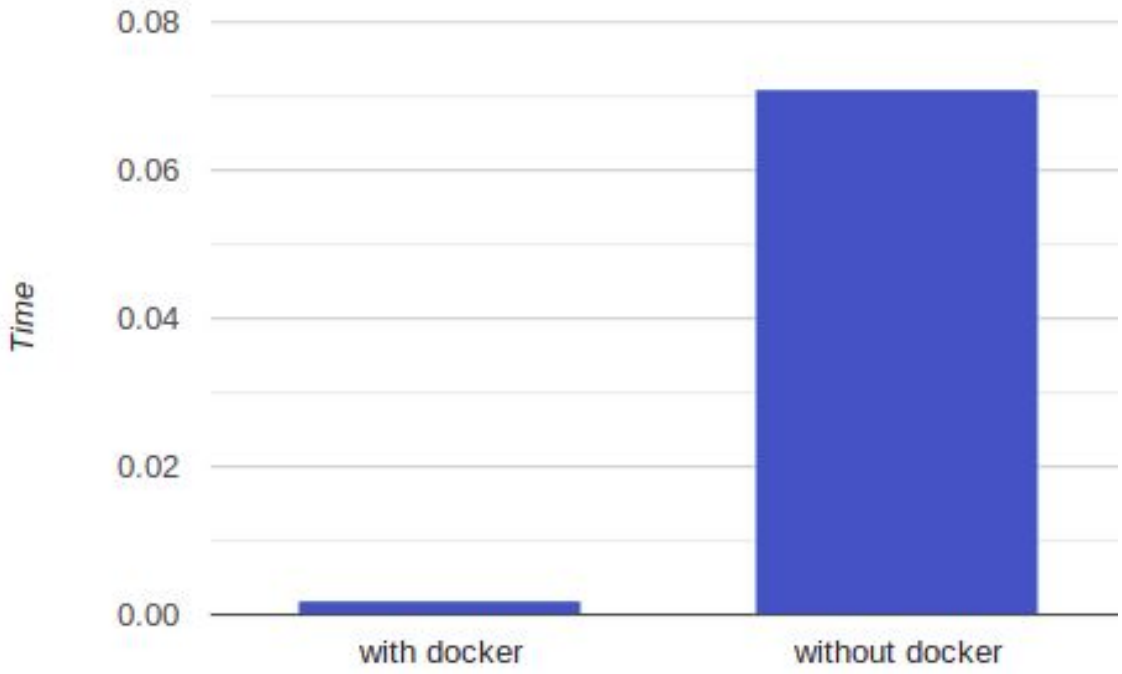
Response time for with docker is way smaller than without docker.

Latency

In **performance testing**, term **latency** of a request is travel time from client to server and server to client. Some tester called it “Delay”.

With Docker: 0.002

Without Docker: 0.071



Inference:

Latency for with docker is way smaller than without docker.

Connect

With Docker: 0.00

Without Docker: 0.00

Inference:

No issues in connectivity for both.

Percentiles

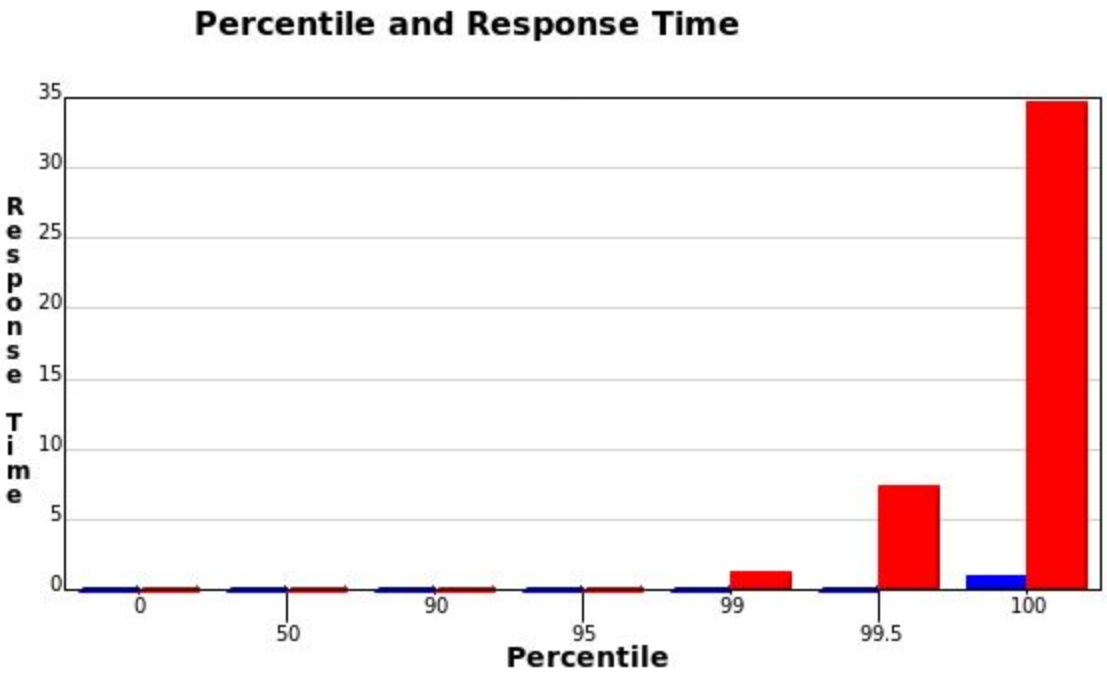
A **percentile** is a measure used in statistics indicating the value below which a given percentage of observations in a group of observations fall. For example, the **response time** for a HTTP request below which 90% of the **response time** values lie, is called the 90-**percentile response time**.

With Docker:

Percentile, %	Resp. Time, s
0.0	0.001
50.0	0.002
90.0	0.003
95.0	0.003
99.0	0.006
99.9	0.013
100.0	1.032

Without Docker:

Percentile, %	Resp. Time, s
0.0	0.001
50.0	0.005
90.0	0.01
95.0	0.017
99.0	1.232
99.9	7.452
100.0	34.688



Inference:

Comparing both with and without docker respectively at each percentile, the response time for with docker is lesser than without docker.

Request label stats

With Docker:

label	status	succ	avg_rt	error
http://172.18.0.1/	OK	100.00%	0.002	

Without Docker:

label	status	succ	avg_rt	error
http://localhost/	FAIL	99.99%	0.073	Non HTTP response message: Socket closed

Hardware

With Docker:

```
     conn-all: 0
   disk-space: 92.600
   bytes-sent: 164,535
  engine-loop: 0.093
   disk-write: 0
          mem: 70.200
   bytes-recv: 166,912
    disk-read: 0
          cpu: 50.500
```

Without Docker:

```
   disk-space: 93.100
   bytes-sent: 4,719
   bytes-recv: 1,839
  engine-loop: 2.696
    disk-read: 2,841
   disk-write: 51,151
          cpu: 26.600
          mem: 83.300
     conn-all: 220
```

Inference:

With docker, less of hardware is used in comparison with without docker.

Conclusion

- Docker comparatively uses less response time and hardware and has higher success rate.
- It enables more efficient use of system resources and enables faster software delivery cycles.
- Docker containers ensure consistency across multiple development and release cycles, thus standardizing your environment.
- It ensures consistent environments from development to production.