**CSE 4074 – Computer Networks Project**

Suleyman Barış Eser – 150116055

Mert İsmail Eği – 150115025

**TABLE OF CONTENTS**

[1. HTTP Server 2](#_Toc61701752)

[2. Proxy Server 7](#_Toc61701753)

[Bonus Part 9](#_Toc61701754)

[If server is not running 10](#_Toc61701755)

[If server running and requested file size is smaller than 10000. 10](#_Toc61701756)

[If requested file size is greater than 10000. 11](#_Toc61701757)

[If same file requested twice 12](#_Toc61701758)

[3. Using ApacheBench (ab) program: 12](#_Toc61701759)

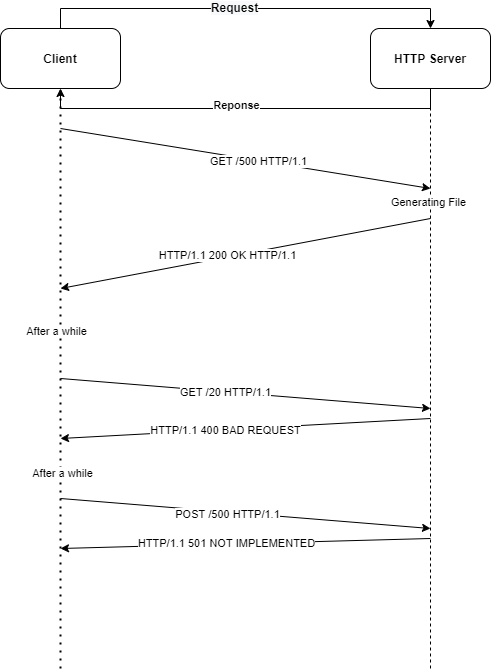
[Outputs 18](#_Toc61701760)

[4. Testing your server using ApacheBench 19](#_Toc61701761)

[HTTP Server 19](#_Toc61701762)

[Proxy 30](#_Toc61701763)

# HTTP Server



Before starting coding, we have to import socket package. Which will help us to do the job. After that, we have to open a socket. To do that, we need an ip address and a port number. Our ip address will be local ip address and port number will be 5000 for now. It will be dynamic on the [1-b section](#HttpServerPartB).

**import** socket  
SERVER = socket.gethostbyname(socket.gethostname())  
ADDR = (SERVER, 5000)  
FORMAT = 'utf-8'  
httpserver = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)  
httpserver.bind(ADDR)

FORMAT variable is our text format for encoding responses and decoding requests.

1. First of all, we have to implement a threading functionality to our code. Because our code should give response to different clients. We used “Threading” library to do this.

**import** threading

we created a function, which will handle requests. Now, we should create a thread for each client (Keep alive: 1).

1. Now, we can take an argument for port number from user.

# python3 httpserver.py PORT\_NUMBER

PORT = int(argv[1])

ADDR = (SERVER, PORT)

1. We can accept client and give a response them.

conn, addr = httpserver.accept()  
thread = threading.Thread(target=handle\_client, args=(conn, addr))  
thread.start()

now, request will be handled in handle\_client function. Firstly, we have to understand request message.

GET /favicon.ico HTTP/1.1  
Host: 192.168.1.102:5000  
Connection: keep-alive  
Pragma: no-cache  
Cache-Control: no-cache  
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/87.0.4280.141 Safari/537.36  
Accept: image/avif,image/webp,image/apng,image/\*,\*/\*;q=0.8  
Referer: http://192.168.1.102:5000/500  
Accept-Encoding: gzip, deflate  
Accept-Language: en-US,en;q=0.9,tr;q=0.8

An example is given above. We can see first line is carrying method, file size and http version. Second line carrying host address. The other lines carrying different properties. After we decode request, we have to split first line into strings.

request = conn.recv(HEADER).decode(FORMAT)  
headers = request.split("\n")  
req\_info = headers[0].split()  
method = req\_info[0] # GET  
url = req\_info[1] # /500  
proto = req\_info[2] # HTTP/1.1

In this method, we handle only GET requests. Therefore, for another request method we have to response 501 Not Implemented message. If request is GET then, we have to detect requested file size, if our file size is greater than and equal to 100 and smaller than or equal to 20000, then we should send back a response message and a file which have size of requested file size.

**if** method == "GET":  
**try**:  
           file\_size = get\_file\_size(url)  
            **if** url == '/favicon.ico':  
                response\_status = "200 OK"  
            **elif** 100 <= file\_size <= 20000:  
                response\_status = "200 OK"  
                content = create\_file(file\_size)  
            **else**:  
                response\_status = "400 Bad Request"  
        **except**:  
            response\_status = "400 Bad Request"  
**else**:  
response\_status = "501 Not Implemented"

In create\_file function, we return a file which contains n letter where n is size of requested file given by client.

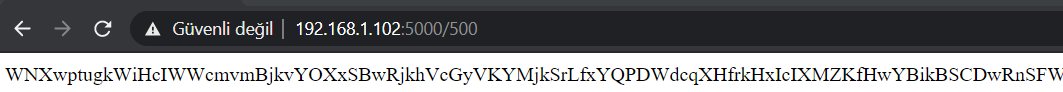
**def** **create\_file**(size):  
    content = ""  
    **for** \_ **in** range(size):  
        lower\_upper\_alphabet = string.ascii\_letters  
        random\_letter = random.choice(lower\_upper\_alphabet)  
        content += random\_letter  
    content = f"<!DOCTYPE html><html><head><title>{str(size)} bytes</title></head><body>{content}</body></html>"  
    **return** content

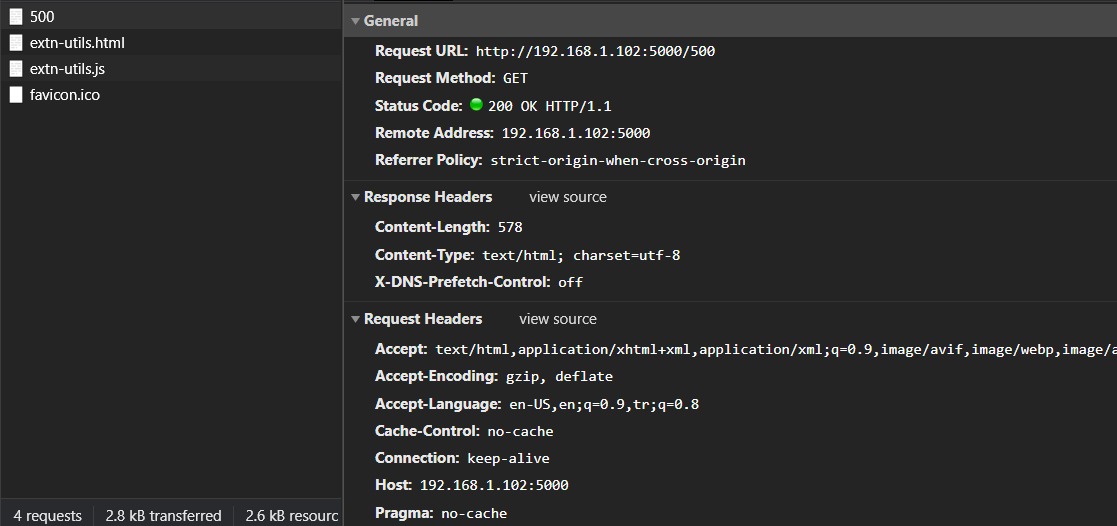
1. Then, values are passing send\_response function. This function is send a response to client.

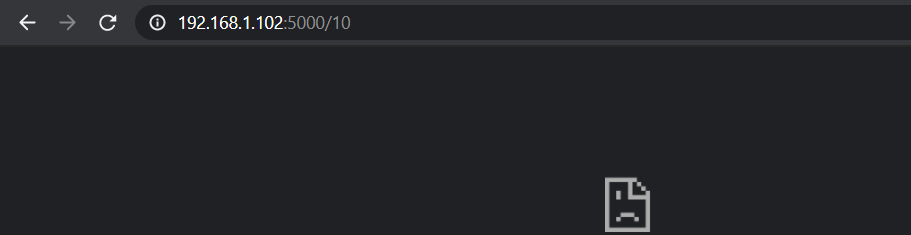
**def** **send\_response**(connection, status, file, proto):  
    response = str(  
        f"{proto} {status} {proto} \r\nContent-Length: {str(len(file))}\r\nContent-Type: {'text/html' **if** file **else** 'x-icon'}; charset={FORMAT}\r\n\r\n")  
    print(  
        f"[SERVER-RESPONSE MESSAGE]\n-------------------------------------\n{response}")  
    connection.sendall(response.encode(FORMAT))  
    connection.sendall(file.encode(FORMAT))

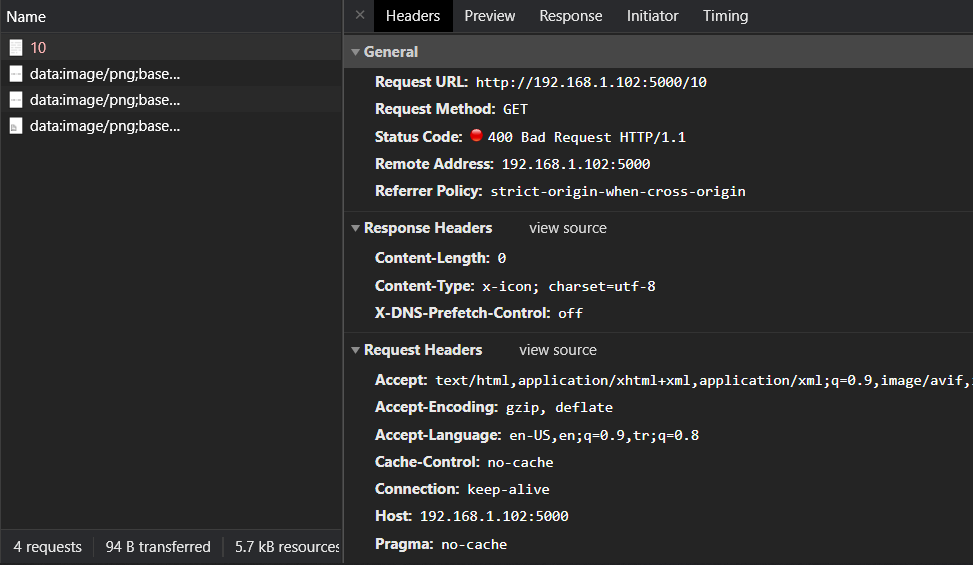
After that function, connection is closing.

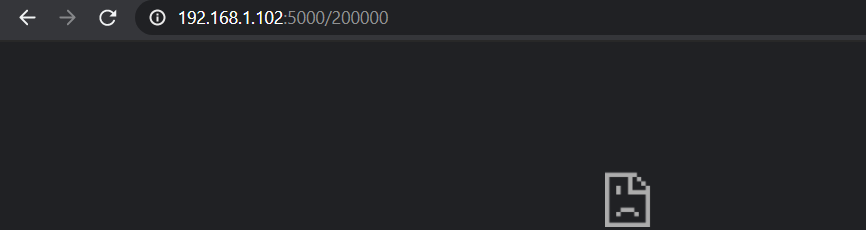
1. Here are same example runs.

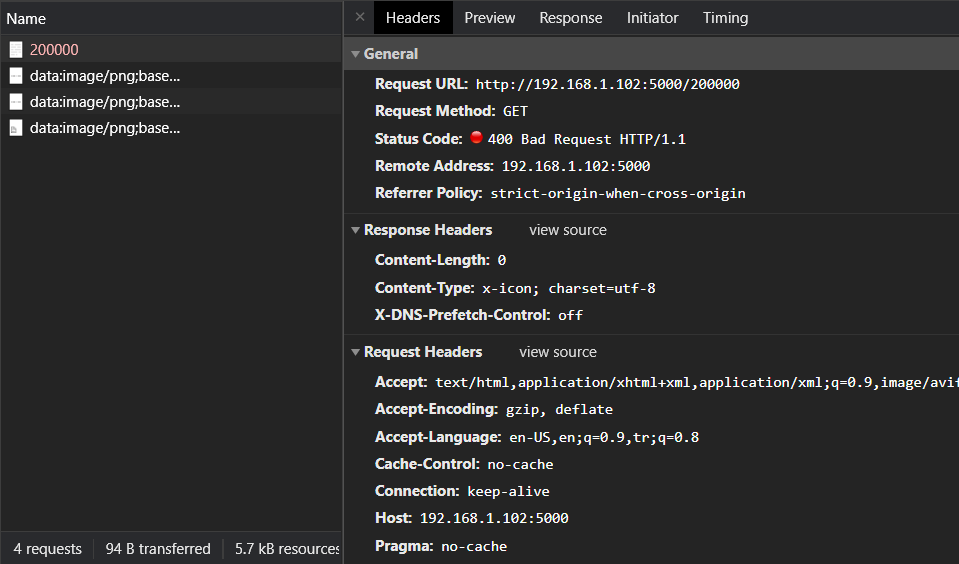








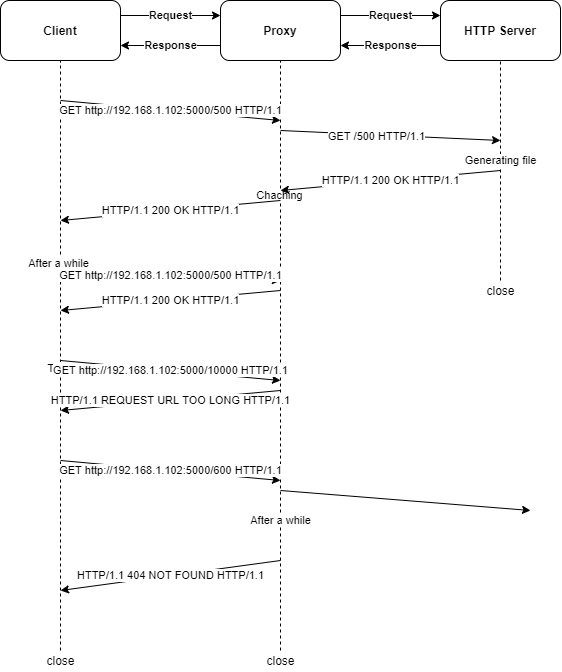




We will not be able to show a different method on browser. But here is a console output.

[SERVER-NEW CONNECTION]  
-------------------------------------   
('192.168.1.102', 56502) **is** connected.  
  
[SERVER-ACTIVE CONNECTIONS] 1  
['PUT', '/2000', 'HTTP/1.0']  
  
[SERVER-REQUEST INFORMATION]  
-------------------------------------  
PUT /2000 HTTP/1.0  
  
File Size= 0  
[SERVER-RESPONSE MESSAGE]  
-------------------------------------  
HTTP/1.0 501 Not Implemented HTTP/1.0  
Content-Length: 0  
Content-Type: x-icon; charset=utf-8

# Proxy Server



1. In Proxy, Handling request is as same as in [Http Server](#HttpServer). In this part, we only redirect client’s request to server then, we redirect httpserver’s response to client.

s.send(request)         # send request to webserver  
**while** 1:  
 # receive data from web server  
 data = s.recv(HEADER)  
 print(f'[CONNECTION] SERVER -> PROXY')  
 # print(f"[MESSAGE] {data}")  
  
 **if** (len(data) > 0):  
 # send to browser  
 conn.send(data)  
 content = data.decode(FORMAT)  
 **if** "Content-Length" **not** **in** content:  
 save\_to\_cache(filename, content)  
 **else**:  
 print(f'[CONNECTION] PROXY -> CLIENT')  
 **else**:  
 print(f'[CONNECTION] PROXY -> CLIENT')  
 print(f"[MESSAGE] NULL")

1. If request is GET method, then request will redirect to httpserver. Otherwise, connection will be closed.

**if** webserver == SERVER **and** url[0] == "GET":

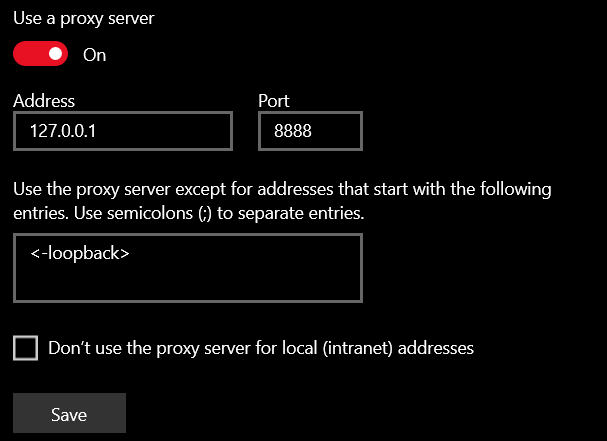
1. This if contidion checks if current request is for our httpserver and it is a GET request. We are not handling other server requests.
2. If requested file size is greater than or equal to 10000 bytes then, proxy send 414 REQUEST TOO LONG response.

**if**(file\_size <= 9999):  
 send\_response(conn, "200 OK", cache\_content, "HTTP/1.1")  
**else**:  
 file = read\_file('/414.html')  
 response = str(  
 f"HTTP/1.1 414 Request-URI Too Long HTTP/1.1 \r\nContent-Length: {str(len(file))}\r\nContent-Type: text/html; charset={FORMAT}\r\n\r\n")  
 print(f'[CONNECTION]\n{response}')  
 conn.sendall(response.encode(FORMAT))

1. In the connect\_server function, we are trying to open a socket connection to httpserver but if connection will end with an error after a timeout then, proxy send a 404 NOT FOUND response.

s = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)  
 **try**:  
 s.connect((webserver, port))  
           print(f'[CONNECTION] PROXY -> SERVER')  
           new\_message = request.decode(FORMAT).split('\n')[0]  
           print(f"[MESSAGE] {new\_message}")  
      **except**:  
           response = str(  
            f"HTTP/1.1 404 Not Found HTTP/1.1 \r\nContent-Length: 0\r\nContent-Type: text/html; charset={FORMAT}\r\n\r\n")  
           print(f'[RESPONSE]\n404 NOT FOUND')  
           conn.sendall(response.encode(FORMAT))  
           s.close()  
           conn.close()

1. Proxy setting are done in Settings>Network & Internet>Proxy.



## Bonus Part

**try**:  
        cached\_file = open(f"cache{filename}", 'r')  
        content = cached\_file.read()  
        cached\_file.close()  
        **if** file\_size == len(content) - 79:  
            **return** content  
        **else**:  
            **return** None  
**except** IOError:  
        print("AN ERROR OCCURED")  
        **return** None

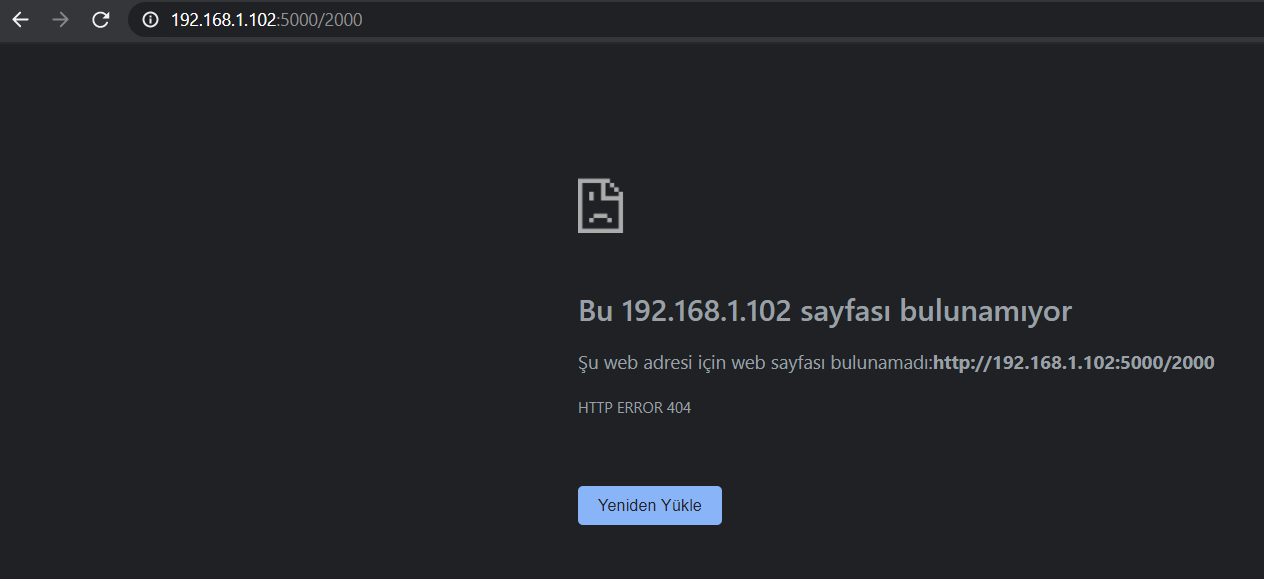
In this function we read the file that is stored in the cache but there is a problem. In concurrent request while a file write a content another one can try to read or write it. This cause a huge problem, to fix this we check file size of cached file and request file size. If both of them are equal than we can read it. Otherwise, we return null.

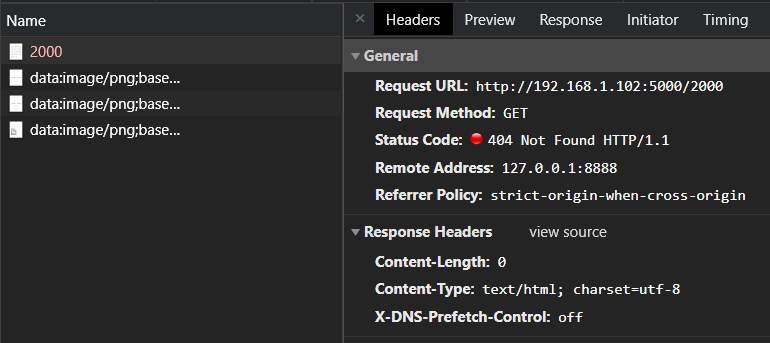
s.send(request)         # send request to webserver  
**while** 1:  
 # receive data from web server  
 data = s.recv(HEADER)  
 # print(f"[MESSAGE] {data}")  
 **if** (len(data) > 0):  
 # send to browser  
 conn.send(data)  
 content = data.decode(FORMAT)  
 **if** "Content-Length" **not** **in** content **and** **not** path.exists(f"cache{filename}"):  
 save\_to\_cache(filename, content)  
 **else**:  
                **break**

In the above code, we get content from httpserver, but there is a problem which is similar with above one. There can be more than one request that try to write on the same file. We have to set a boundry. If a thread create a file and start to write onto it then, no thread can access it neither for writing nor for reading.

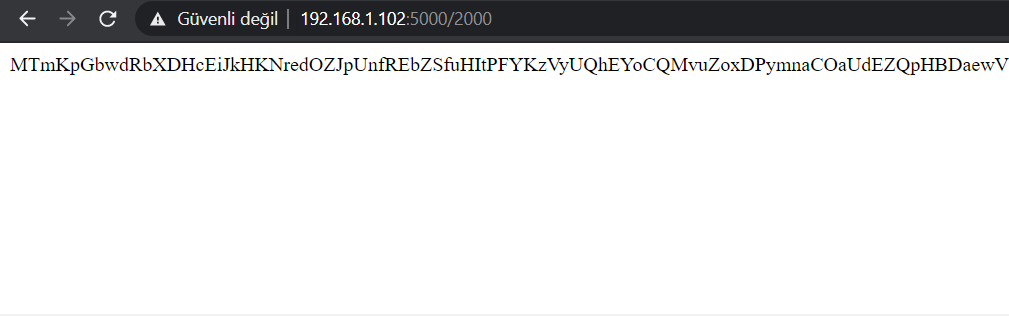
Here is sample runs:

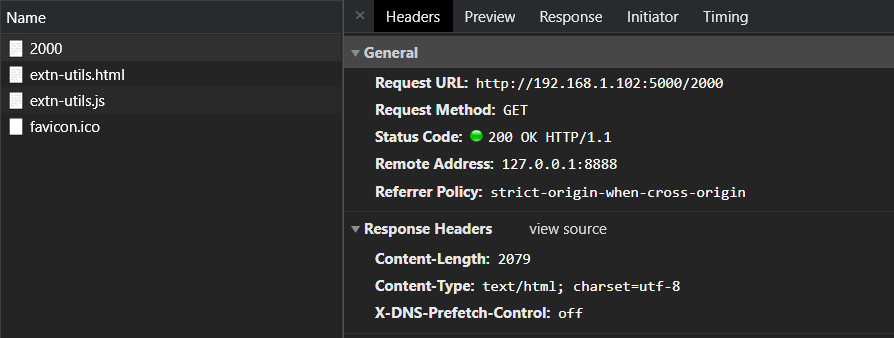
## If server is not running



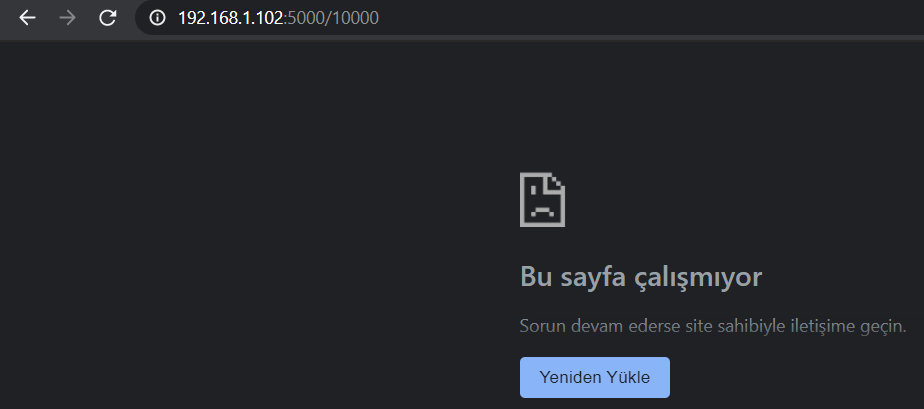


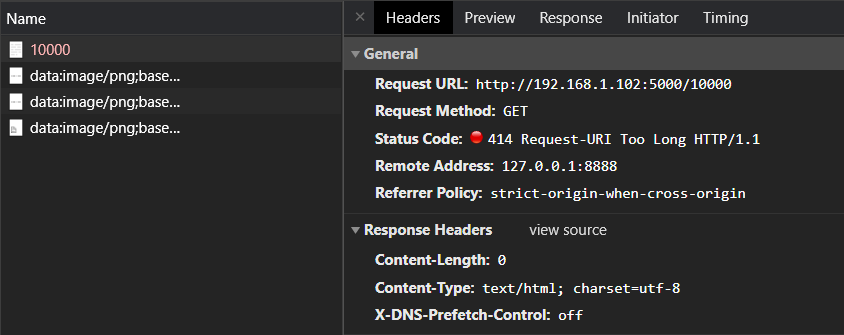
## If server running and requested file size is smaller than 10000.



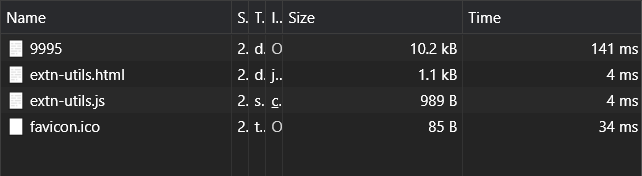


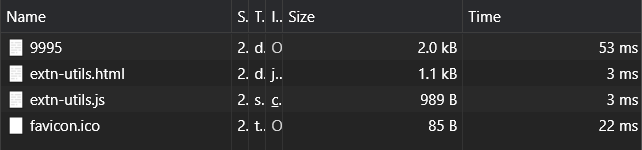
## If requested file size is greater than 10000.





## If same file requested twice

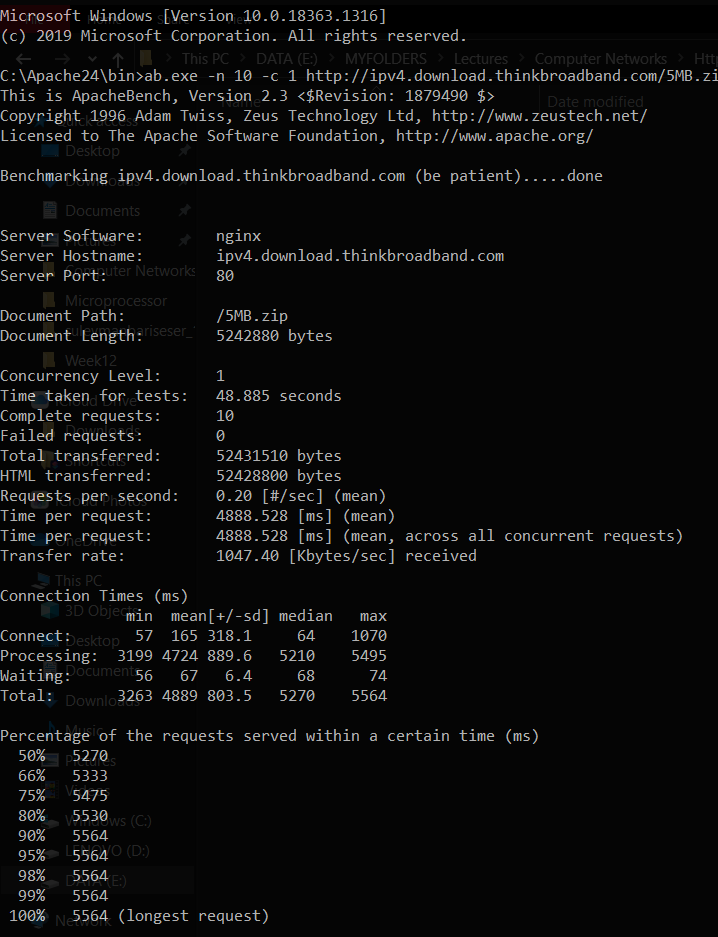




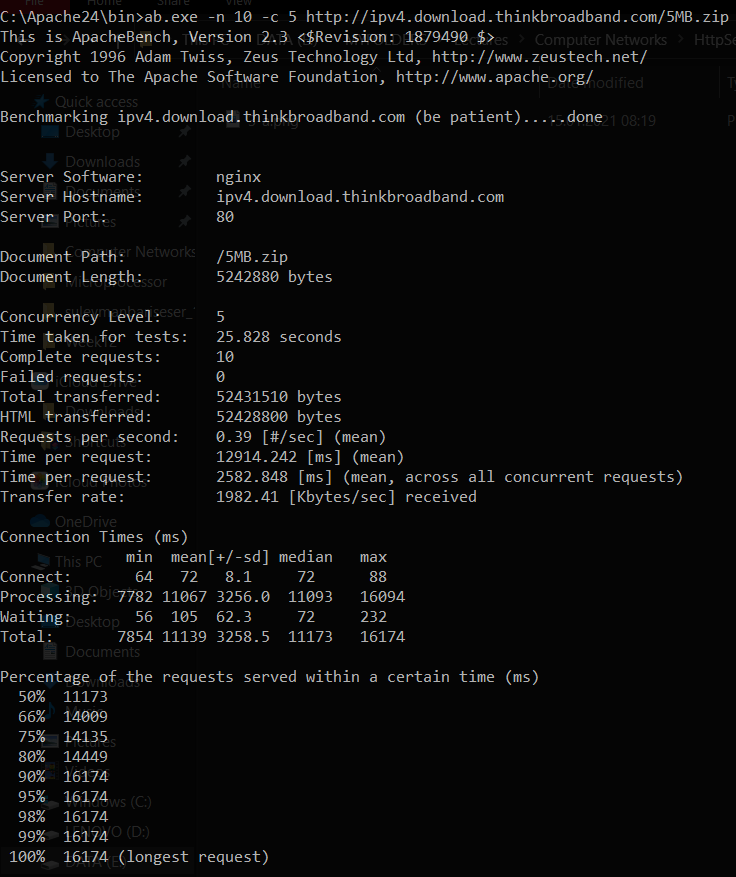
We can see time is almost 1/3 of first request.

# Using ApacheBench (ab) program:

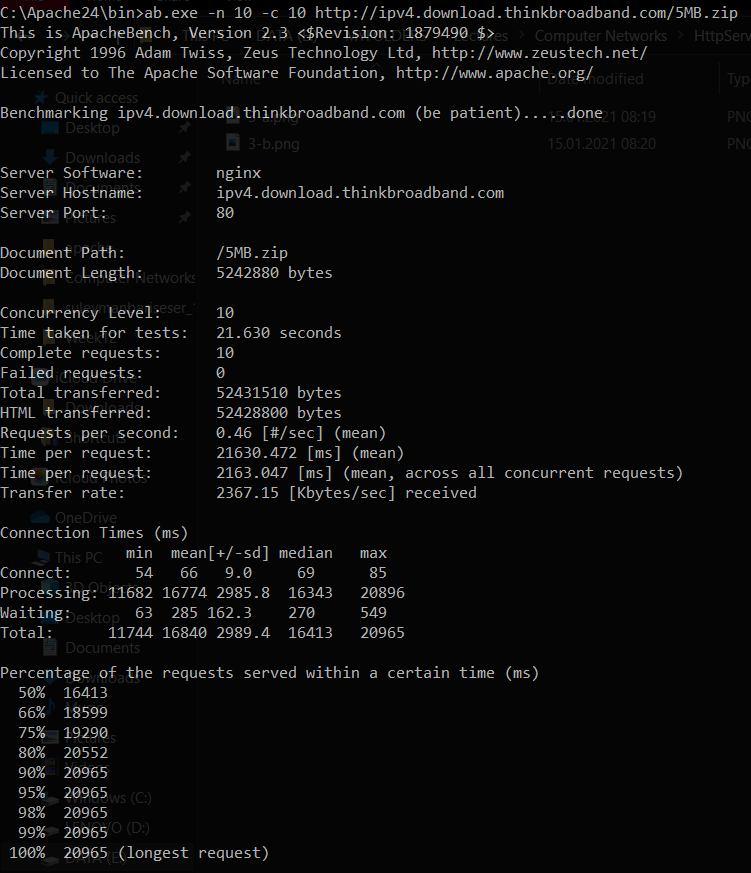
ab –n 10 –c 1 http://ipv4.download.thinkbroadband.com/5MB.zip

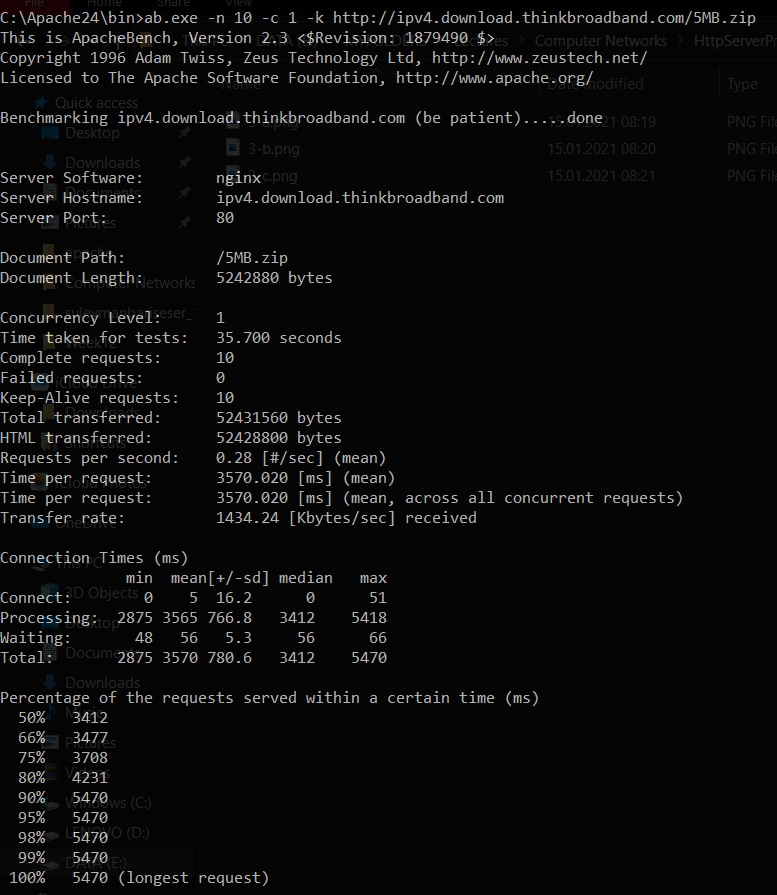


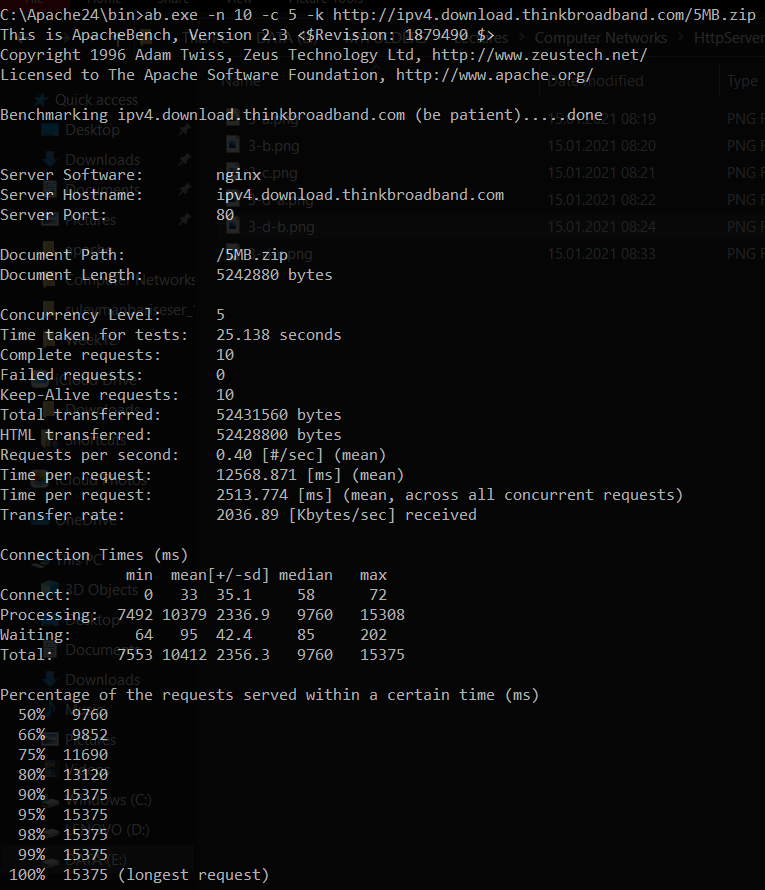
ab –n 10 –c 5 http://ipv4.download.thinkbroadband.com/5MB.zip

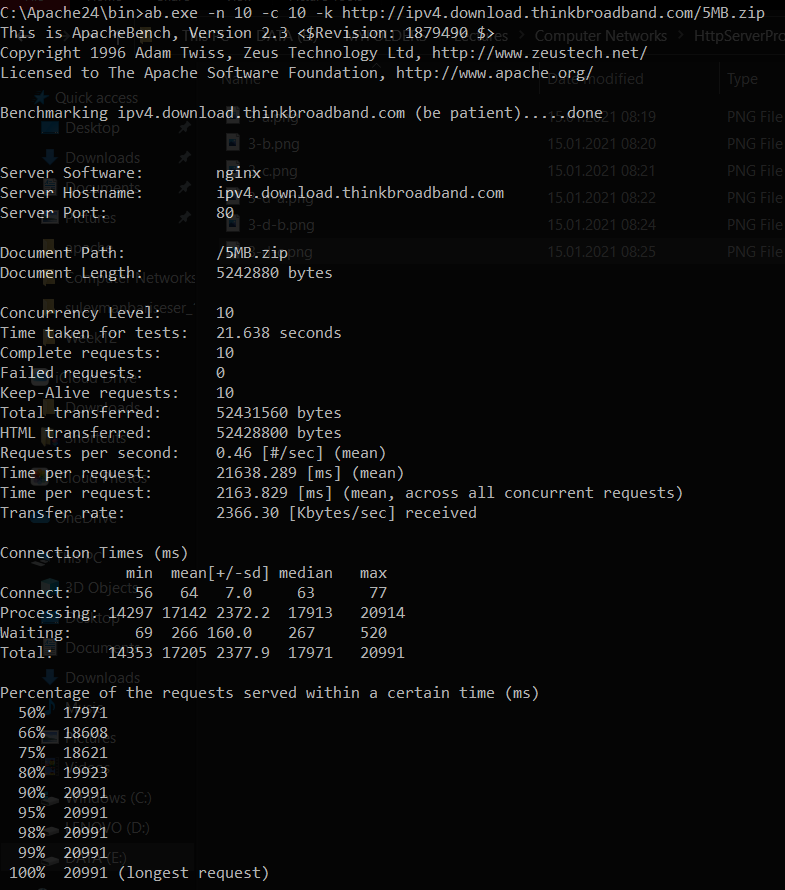


ab –n 10 –c 10 http://ipv4.download.thinkbroadband.com/5MB.zip









## Outputs

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Labels | 3-a | 3-b | 3-c | 3-d-a | 3-d-b | 3-d-c |
| Time taken for tests(seconds) | 48.885 | 25.828 | 21.630 | 35.700 | 25.138 | 21.638 |
| Total Transferred | 52431510 | 52431510 | 52431510 | 52431510 | 52431560 | 52431560 |
| HTML transferred | 52428800 | 52428800 | 52428800 | 52428800 | 52428800 | 52428800 |
| Time per Request | 4888.528 | 2582.848 | 2163.047 | 3570.020 | 2513.774 | 2163.829 |
| Requests per second | 0.20 | 0.39 | 0.46 | 0.28 | 0.40 | 0.46 |
| Transfer Rate | 1047.40 | 1982.41 | 2367.15 | 1434.24 | 2036.89 | 2366.30 |

We send same request for 10 times with different concurrency level.

1. From first row of table, we can see time taken for tests is decreased while concurrency level is increasing. For small concurrency level, when we add -k parameter, difference between them is huge but for high concurreny level, differences are small.
2. Size of transferred files do not change. Because we always request same file. Same for HTML transferred.
3. Time per request has same attitude with time taken for tests. If we use low concurrency levels than we have to wait more.
4. Same with above but reverse direction. If we increase concurrency level then, we have high requests per second.
5. When concurrency level is high, we have higher transfer rates.
6. Connection times generally increase when concurreny level is increasing

Main reason of them, when there is a concurrence request, there will be a queue and this will cause connection times will increase. But when concurrency is high then, we use more source in a unit time so, our request will be responded in shorter time. Our number of request does not change, but our response time decreases so time per request will decrease. We have more request in a unit time, this will increase our requests per second and transfer rate.

# Testing your server using ApacheBench

## HTTP Server

C:\Apache24\bin>ab -n 100 -c 1 http://192.168.1.102:5000/9000  
This **is** ApacheBench, Version 2.3 <$Revision: 1879490 $>  
Copyright 1996 Adam Twiss, Zeus Technology Ltd, http://www.zeustech.net/  
Licensed to The Apache Software Foundation, http://www.apache.org/  
  
Benchmarking 192.168.1.102 (be patient).....done  
  
  
Server Software:  
Server Hostname: 192.168.1.102  
Server Port: 5000  
  
Document Path: /9000  
Document Length: 9079 bytes  
  
Concurrency Level: 1  
Time taken **for** tests: 1.692 seconds  
Complete requests: 100  
Failed requests: 0  
Total transferred: 917000 bytes  
HTML transferred: 907900 bytes  
Requests per second: 59.12 [#/sec] (mean)  
Time per request: 16.916 [ms] (mean)  
Time per request: 16.916 [ms] (mean, across all concurrent requests)  
Transfer rate: 529.40 [Kbytes/sec] received  
  
Connection Times (ms)  
 min mean[+/-sd] median max  
Connect: 0 0 1.2 0 10  
Processing: 12 16 2.2 16 29  
Waiting: 12 16 2.2 16 29  
Total: 12 17 2.5 16 29  
  
Percentage of the requests served within a certain time (ms)  
 50% 16  
 66% 16  
 75% 16  
 80% 17  
 90% 20  
 95% 20  
 98% 28  
 99% 29  
 100% 29 (longest request)

C:\Apache24\bin>ab -n 100 -c 5 http://192.168.1.102:5000/9000  
This **is** ApacheBench, Version 2.3 <$Revision: 1879490 $>  
Copyright 1996 Adam Twiss, Zeus Technology Ltd, http://www.zeustech.net/  
Licensed to The Apache Software Foundation, http://www.apache.org/  
  
Benchmarking 192.168.1.102 (be patient).....done  
  
  
Server Software:  
Server Hostname: 192.168.1.102  
Server Port: 5000  
  
Document Path: /9000  
Document Length: 9079 bytes  
  
Concurrency Level: 5  
Time taken **for** tests: 1.750 seconds  
Complete requests: 100  
Failed requests: 0  
Total transferred: 917000 bytes  
HTML transferred: 907900 bytes  
Requests per second: 57.13 [#/sec] (mean)  
Time per request: 87.517 [ms] (mean)  
Time per request: 17.503 [ms] (mean, across all concurrent requests)  
Transfer rate: 511.62 [Kbytes/sec] received  
  
Connection Times (ms)  
 min mean[+/-sd] median max  
Connect: 0 0 0.8 0 4  
Processing: 27 85 27.8 83 160  
Waiting: 27 84 27.8 82 160  
Total: 27 85 27.7 83 160  
  
Percentage of the requests served within a certain time (ms)  
 50% 83  
 66% 94  
 75% 100  
 80% 108  
 90% 124  
 95% 136  
 98% 152  
 99% 160  
 100% 160 (longest request)

C:\Apache24\bin>ab -n 100 -c 10 http://192.168.1.102:5000/9000  
This **is** ApacheBench, Version 2.3 <$Revision: 1879490 $>  
Copyright 1996 Adam Twiss, Zeus Technology Ltd, http://www.zeustech.net/  
Licensed to The Apache Software Foundation, http://www.apache.org/  
  
Benchmarking 192.168.1.102 (be patient).....done  
  
  
Server Software:  
Server Hostname: 192.168.1.102  
Server Port: 5000  
  
Document Path: /9000  
Document Length: 9079 bytes  
  
Concurrency Level: 10  
Time taken **for** tests: 1.930 seconds  
Complete requests: 100  
Failed requests: 0  
Total transferred: 917000 bytes  
HTML transferred: 907900 bytes  
Requests per second: 51.82 [#/sec] (mean)  
Time per request: 192.965 [ms] (mean)  
Time per request: 19.297 [ms] (mean, across all concurrent requests)  
Transfer rate: 464.08 [Kbytes/sec] received  
  
Connection Times (ms)  
 min mean[+/-sd] median max  
Connect: 0 0 0.9 0 4  
Processing: 46 182 47.8 184 316  
Waiting: 42 181 47.5 184 316  
Total: 46 182 47.8 184 316  
  
Percentage of the requests served within a certain time (ms)  
 50% 184  
 66% 196  
 75% 212  
 80% 217  
 90% 236  
 95% 268  
 98% 284  
 99% 316  
 100% 316 (longest request)

C:\Apache24\bin>ab -n 100 -c 50 http://192.168.1.102:5000/9000  
This **is** ApacheBench, Version 2.3 <$Revision: 1879490 $>  
Copyright 1996 Adam Twiss, Zeus Technology Ltd, http://www.zeustech.net/  
Licensed to The Apache Software Foundation, http://www.apache.org/  
  
Benchmarking 192.168.1.102 (be patient).....done  
  
  
Server Software:  
Server Hostname: 192.168.1.102  
Server Port: 5000  
  
Document Path: /9000  
Document Length: 9079 bytes  
  
Concurrency Level: 50  
Time taken **for** tests: 4.298 seconds  
Complete requests: 100  
Failed requests: 0  
Total transferred: 917000 bytes  
HTML transferred: 907900 bytes  
Requests per second: 23.27 [#/sec] (mean)  
Time per request: 2148.987 [ms] (mean)  
Time per request: 42.980 [ms] (mean, across all concurrent requests)  
Transfer rate: 208.36 [Kbytes/sec] received  
  
Connection Times (ms)  
 min mean[+/-sd] median max  
Connect: 0 41 139.0 0 527  
Processing: 34 1626 757.4 2058 2583  
Waiting: 28 1194 656.0 1022 2070  
Total: 66 1667 759.2 2058 2585  
  
Percentage of the requests served within a certain time (ms)  
 50% 2058  
 66% 2067  
 75% 2074  
 80% 2075  
 90% 2576  
 95% 2578  
 98% 2583  
 99% 2585  
 100% 2585 (longest request)

C:\Apache24\bin>ab -n 100 -c 100 http://192.168.1.102:5000/9000  
This **is** ApacheBench, Version 2.3 <$Revision: 1879490 $>  
Copyright 1996 Adam Twiss, Zeus Technology Ltd, http://www.zeustech.net/  
Licensed to The Apache Software Foundation, http://www.apache.org/  
  
Benchmarking 192.168.1.102 (be patient).....done  
  
  
Server Software:  
Server Hostname: 192.168.1.102  
Server Port: 5000  
  
Document Path: /9000  
Document Length: 9079 bytes  
  
Concurrency Level: 100  
Time taken **for** tests: 4.324 seconds  
Complete requests: 100  
Failed requests: 0  
Total transferred: 917000 bytes  
HTML transferred: 907900 bytes  
Requests per second: 23.12 [#/sec] (mean)  
Time per request: 4324.337 [ms] (mean)  
Time per request: 43.243 [ms] (mean, across all concurrent requests)  
Transfer rate: 207.09 [Kbytes/sec] received  
  
Connection Times (ms)  
 min mean[+/-sd] median max  
Connect: 0 41 138.2 0 518  
Processing: 25 2090 1309.5 2063 4110  
Waiting: 25 2042 1306.0 2054 4076  
Total: 49 2130 1307.7 2063 4110  
  
Percentage of the requests served within a certain time (ms)  
 50% 2063  
 66% 3076  
 75% 3077  
 80% 3581  
 90% 4088  
 95% 4088  
 98% 4088  
 99% 4110  
 100% 4110 (longest request)

C:\Apache24\bin>ab -n 100 -c 1 -k http://192.168.1.102:5000/9000  
This **is** ApacheBench, Version 2.3 <$Revision: 1879490 $>  
Copyright 1996 Adam Twiss, Zeus Technology Ltd, http://www.zeustech.net/  
Licensed to The Apache Software Foundation, http://www.apache.org/  
  
Benchmarking 192.168.1.102 (be patient).....done  
  
  
Server Software:  
Server Hostname: 192.168.1.102  
Server Port: 5000  
  
Document Path: /9000  
Document Length: 9079 bytes  
  
Concurrency Level: 1  
Time taken **for** tests: 1.778 seconds  
Complete requests: 100  
Failed requests: 0  
Keep-Alive requests: 0  
Total transferred: 917000 bytes  
HTML transferred: 907900 bytes  
Requests per second: 56.24 [#/sec] (mean)  
Time per request: 17.781 [ms] (mean)  
Time per request: 17.781 [ms] (mean, across all concurrent requests)  
Transfer rate: 503.63 [Kbytes/sec] received  
  
Connection Times (ms)  
 min mean[+/-sd] median max  
Connect: 0 0 1.0 0 4  
Processing: 12 17 5.1 16 48  
Waiting: 12 17 5.0 16 48  
Total: 12 18 5.0 16 48  
  
Percentage of the requests served within a certain time (ms)  
 50% 16  
 66% 16  
 75% 16  
 80% 20  
 90% 20  
 95% 28  
 98% 40  
 99% 48  
 100% 48 (longest request)

C:\Apache24\bin>ab -n 100 -c 5 -k http://192.168.1.102:5000/9000  
This **is** ApacheBench, Version 2.3 <$Revision: 1879490 $>  
Copyright 1996 Adam Twiss, Zeus Technology Ltd, http://www.zeustech.net/  
Licensed to The Apache Software Foundation, http://www.apache.org/  
  
Benchmarking 192.168.1.102 (be patient).....done  
  
  
Server Software:  
Server Hostname: 192.168.1.102  
Server Port: 5000  
  
Document Path: /9000  
Document Length: 9079 bytes  
  
Concurrency Level: 5  
Time taken **for** tests: 1.986 seconds  
Complete requests: 100  
Failed requests: 0  
Keep-Alive requests: 0  
Total transferred: 917000 bytes  
HTML transferred: 907900 bytes  
Requests per second: 50.35 [#/sec] (mean)  
Time per request: 99.305 [ms] (mean)  
Time per request: 19.861 [ms] (mean, across all concurrent requests)  
Transfer rate: 450.89 [Kbytes/sec] received  
  
Connection Times (ms)  
 min mean[+/-sd] median max  
Connect: 0 0 0.9 0 4  
Processing: 32 96 32.1 93 200  
Waiting: 32 96 32.1 93 200  
Total: 32 97 32.1 96 200  
  
Percentage of the requests served within a certain time (ms)  
 50% 96  
 66% 109  
 75% 116  
 80% 116  
 90% 144  
 95% 160  
 98% 180  
 99% 200  
 100% 200 (longest request)

C:\Apache24\bin>ab -n 100 -c 10 -k http://192.168.1.102:5000/9000  
This **is** ApacheBench, Version 2.3 <$Revision: 1879490 $>  
Copyright 1996 Adam Twiss, Zeus Technology Ltd, http://www.zeustech.net/  
Licensed to The Apache Software Foundation, http://www.apache.org/  
  
Benchmarking 192.168.1.102 (be patient).....done  
  
  
Server Software:  
Server Hostname: 192.168.1.102  
Server Port: 5000  
  
Document Path: /9000  
Document Length: 9079 bytes  
  
Concurrency Level: 10  
Time taken **for** tests: 1.996 seconds  
Complete requests: 100  
Failed requests: 0  
Keep-Alive requests: 0  
Total transferred: 917000 bytes  
HTML transferred: 907900 bytes  
Requests per second: 50.09 [#/sec] (mean)  
Time per request: 199.623 [ms] (mean)  
Time per request: 19.962 [ms] (mean, across all concurrent requests)  
Transfer rate: 448.60 [Kbytes/sec] received  
  
Connection Times (ms)  
 min mean[+/-sd] median max  
Connect: 0 0 0.7 0 4  
Processing: 28 189 51.9 194 313  
Waiting: 28 189 51.9 192 313  
Total: 28 190 51.8 194 313  
  
Percentage of the requests served within a certain time (ms)  
 50% 194  
 66% 212  
 75% 220  
 80% 229  
 90% 251  
 95% 280  
 98% 296  
 99% 313  
 100% 313 (longest request)

C:\Apache24\bin>ab -n 100 -c 50 -k http://192.168.1.102:5000/9000  
This **is** ApacheBench, Version 2.3 <$Revision: 1879490 $>  
Copyright 1996 Adam Twiss, Zeus Technology Ltd, http://www.zeustech.net/  
Licensed to The Apache Software Foundation, http://www.apache.org/  
  
Benchmarking 192.168.1.102 (be patient).....done  
  
  
Server Software:  
Server Hostname: 192.168.1.102  
Server Port: 5000  
  
Document Path: /9000  
Document Length: 9079 bytes  
  
Concurrency Level: 50  
Time taken **for** tests: 4.342 seconds  
Complete requests: 100  
Failed requests: 0  
Keep-Alive requests: 0  
Total transferred: 917000 bytes  
HTML transferred: 907900 bytes  
Requests per second: 23.03 [#/sec] (mean)  
Time per request: 2171.245 [ms] (mean)  
Time per request: 43.425 [ms] (mean, across all concurrent requests)  
Transfer rate: 206.22 [Kbytes/sec] received  
  
Connection Times (ms)  
 min mean[+/-sd] median max  
Connect: 0 41 139.4 0 520  
Processing: 52 1640 778.5 2065 2597  
Waiting: 27 1192 658.9 1046 2082  
Total: 52 1681 779.9 2065 2599  
  
Percentage of the requests served within a certain time (ms)  
 50% 2065  
 66% 2075  
 75% 2079  
 80% 2571  
 90% 2583  
 95% 2595  
 98% 2597  
 99% 2599  
 100% 2599 (longest request)

C:\Apache24\bin>ab -n 100 -c 100 -k http://192.168.1.102:5000/9000  
This **is** ApacheBench, Version 2.3 <$Revision: 1879490 $>  
Copyright 1996 Adam Twiss, Zeus Technology Ltd, http://www.zeustech.net/  
Licensed to The Apache Software Foundation, http://www.apache.org/  
  
Benchmarking 192.168.1.102 (be patient).....done  
  
  
Server Software:  
Server Hostname: 192.168.1.102  
Server Port: 5000  
  
Document Path: /9000  
Document Length: 9079 bytes  
  
Concurrency Level: 100  
Time taken **for** tests: 4.339 seconds  
Complete requests: 100  
Failed requests: 0  
Keep-Alive requests: 0  
Total transferred: 917000 bytes  
HTML transferred: 907900 bytes  
Requests per second: 23.04 [#/sec] (mean)  
Time per request: 4339.438 [ms] (mean)  
Time per request: 43.394 [ms] (mean, across all concurrent requests)  
Transfer rate: 206.36 [Kbytes/sec] received  
  
Connection Times (ms)  
 min mean[+/-sd] median max  
Connect: 0 41 138.3 0 516  
Processing: 77 2144 1320.5 2071 4151  
Waiting: 31 2096 1316.1 2064 4104  
Total: 78 2185 1317.7 2071 4151  
  
Percentage of the requests served within a certain time (ms)  
 50% 2071  
 66% 3097  
 75% 3607  
 80% 3608  
 90% 4120  
 95% 4120  
 98% 4120  
 99% 4151  
 100% 4151 (longest request)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Labels | -c 1 | -c 5 | -c 10 | -c 50 | -c 100 |
| Time taken for tests(seconds) | 1.692 | 1.750 | 1.930 | 4.298 | 4.324 |
| Total Transferred | 917000 | 917000 | 917000 | 917000 | 917000 |
| HTML transferred | 907900 | 907900 | 907900 | 907900 | 907900 |
| Time per Request | 16.916 | 17.503 | 19.297 | 42.980 | 43.243 |
| Requests per second | 59.12 | 57.13 | 51.82 | 23.27 | 23.12 |
| Transfer Rate | 529.40 | 511.62 | 464.08 | 208.36 | 207.09 |

*\* Our server is located on our local device. Therefore, there is no much different when we increase concurrency level by 1 so, we use 1,5,10,50,100 concurrency levels.*

We can see when concurrency level increased, our connections are getting to slower. Main reason is that we create a file which contains n bytes word where n is entered by user. This process takes some time so, when we increase concurrency levels, our transfer rate, time taken for tests and time per request are increasing and request per second is decreasing. Same for below.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Labels | -c 1 -k | -c 5 -k | -c 10 -k | -c 50 -k | -c 100 -k |
| Time taken for tests(seconds) | 1.778 | 1.986 | 1.996 | 4.342 | 4.339 |
| Total Transferred | 917000 | 917000 | 917000 | 917000 | 917000 |
| HTML transferred | 907900 | 907900 | 907900 | 907900 | 907900 |
| Time per Request | 17.781 | 19.861 | 19.962 | 43.425 | 43.394 |
| Requests per second | 56.24 | 50.35 | 50.09 | 23.03 | 23.04 |
| Transfer Rate | 503.63 | 450.89 | 448.60 | 206.22 | 206.36 |

## Proxy

C:\Apache24\bin>**ab -n 100 -c 1 -X 127.0.0.1:8888 http://192.168.1.102:5000/9000**  
This **is** ApacheBench, Version 2.3 <$Revision: 1879490 $>  
Copyright 1996 Adam Twiss, Zeus Technology Ltd, http://www.zeustech.net/  
Licensed to The Apache Software Foundation, http://www.apache.org/  
  
Benchmarking 192.168.1.102 [through 127.0.0.1:8888] (be patient).....done  
  
  
Server Software:  
Server Hostname: 192.168.1.102  
Server Port: 5000  
  
Document Path: /9000  
Document Length: 9079 bytes  
  
Concurrency Level: 1  
Time taken **for** tests: 0.734 seconds  
Complete requests: 100  
Failed requests: 0  
Total transferred: 917000 bytes  
HTML transferred: 907900 bytes  
Requests per second: 136.29 [#/sec] (mean)  
Time per request: 7.337 [ms] (mean)  
Time per request: 7.337 [ms] (mean, across all concurrent requests)  
Transfer rate: 1220.49 [Kbytes/sec] received  
  
Connection Times (ms)  
 min mean[+/-sd] median max  
Connect: 0 0 1.0 0 4  
Processing: 0 7 17.3 4 173  
Waiting: 0 5 5.5 4 42  
Total: 0 7 17.3 4 173  
  
Percentage of the requests served within a certain time (ms)  
 50% 4  
 66% 8  
 75% 8  
 80% 8  
 90% 12  
 95% 16  
 98% 19  
 99% 173  
 100% 173 (longest request)

C:\Apache24\bin>**ab -n 100 -c 5 -X 127.0.0.1:8888 http://192.168.1.102:5000/9000**  
This **is** ApacheBench, Version 2.3 <$Revision: 1879490 $>  
Copyright 1996 Adam Twiss, Zeus Technology Ltd, http://www.zeustech.net/  
Licensed to The Apache Software Foundation, http://www.apache.org/  
  
Benchmarking 192.168.1.102 [through 127.0.0.1:8888] (be patient).....done  
  
  
Server Software:  
Server Hostname: 192.168.1.102  
Server Port: 5000  
  
Document Path: /9000  
Document Length: 9079 bytes  
  
Concurrency Level: 5  
Time taken **for** tests: 0.324 seconds  
Complete requests: 100  
Failed requests: 0  
Total transferred: 917000 bytes  
HTML transferred: 907900 bytes  
Requests per second: 308.35 [#/sec] (mean)  
Time per request: 16.216 [ms] (mean)  
Time per request: 3.243 [ms] (mean, across all concurrent requests)  
Transfer rate: 2761.25 [Kbytes/sec] received  
  
Connection Times (ms)  
 min mean[+/-sd] median max  
Connect: 0 0 1.0 0 4  
Processing: 4 15 4.3 16 28  
Waiting: 4 15 4.3 16 28  
Total: 4 15 4.2 16 28  
  
Percentage of the requests served within a certain time (ms)  
 50% 16  
 66% 16  
 75% 16  
 80% 20  
 90% 20  
 95% 24  
 98% 25  
 99% 28  
 100% 28 (longest request)

C:\Apache24\bin>**ab -n 100 -c 10 -X 127.0.0.1:8888 http://192.168.1.102:5000/9000**  
This **is** ApacheBench, Version 2.3 <$Revision: 1879490 $>  
Copyright 1996 Adam Twiss, Zeus Technology Ltd, http://www.zeustech.net/  
Licensed to The Apache Software Foundation, http://www.apache.org/  
  
Benchmarking 192.168.1.102 [through 127.0.0.1:8888] (be patient).....done  
  
  
Server Software:  
Server Hostname: 192.168.1.102  
Server Port: 5000  
  
Document Path: /9000  
Document Length: 9079 bytes  
  
Concurrency Level: 10  
Time taken **for** tests: 0.318 seconds  
Complete requests: 100  
Failed requests: 0  
Total transferred: 917000 bytes  
HTML transferred: 907900 bytes  
Requests per second: 314.84 [#/sec] (mean)  
Time per request: 31.762 [ms] (mean)  
Time per request: 3.176 [ms] (mean, across all concurrent requests)  
Transfer rate: 2819.40 [Kbytes/sec] received  
  
Connection Times (ms)  
 min mean[+/-sd] median max  
Connect: 0 0 0.7 0 4  
Processing: 12 29 9.0 28 48  
Waiting: 12 29 9.1 28 48  
Total: 12 29 9.1 28 48  
  
Percentage of the requests served within a certain time (ms)  
 50% 28  
 66% 32  
 75% 36  
 80% 36  
 90% 44  
 95% 48  
 98% 48  
 99% 48  
 100% 48 (longest request)

C:\Apache24\bin>**ab -n 100 -c 50 -X 127.0.0.1:8888 http://192.168.1.102:5000/9000**  
This **is** ApacheBench, Version 2.3 <$Revision: 1879490 $>  
Copyright 1996 Adam Twiss, Zeus Technology Ltd, http://www.zeustech.net/  
Licensed to The Apache Software Foundation, http://www.apache.org/  
  
Benchmarking 192.168.1.102 [through 127.0.0.1:8888] (be patient).....done  
  
  
Server Software:  
Server Hostname: 192.168.1.102  
Server Port: 5000  
  
Document Path: /9000  
Document Length: 9079 bytes  
  
Concurrency Level: 50  
Time taken **for** tests: 3.653 seconds  
Complete requests: 100  
Failed requests: 0  
Total transferred: 917000 bytes  
HTML transferred: 907900 bytes  
Requests per second: 27.38 [#/sec] (mean)  
Time per request: 1826.342 [ms] (mean)  
Time per request: 36.527 [ms] (mean, across all concurrent requests)  
Transfer rate: 245.16 [Kbytes/sec] received  
  
Connection Times (ms)  
 min mean[+/-sd] median max  
Connect: 0 36 130.7 0 520  
Processing: 22 1515 540.2 1556 2071  
Waiting: 13 968 454.0 1034 1559  
Total: 515 1551 529.2 1556 2075  
  
Percentage of the requests served within a certain time (ms)  
 50% 1556  
 66% 2062  
 75% 2063  
 80% 2064  
 90% 2070  
 95% 2071  
 98% 2071  
 99% 2075  
 100% 2075 (longest request)

C:\Apache24\bin>**ab -n 100 -c 100 -X 127.0.0.1:8888 http://192.168.1.102:5000/9000**This **is** ApacheBench, Version 2.3 <$Revision: 1879490 $>  
Copyright 1996 Adam Twiss, Zeus Technology Ltd, http://www.zeustech.net/  
Licensed to The Apache Software Foundation, http://www.apache.org/  
  
Benchmarking 192.168.1.102 [through 127.0.0.1:8888] (be patient).....done  
  
  
Server Software:  
Server Hostname: 192.168.1.102  
Server Port: 5000  
  
Document Path: /9000  
Document Length: 9079 bytes  
  
Concurrency Level: 100  
Time taken **for** tests: 3.663 seconds  
Complete requests: 100  
Failed requests: 0  
Total transferred: 917000 bytes  
HTML transferred: 907900 bytes  
Requests per second: 27.30 [#/sec] (mean)  
Time per request: 3662.762 [ms] (mean)  
Time per request: 36.628 [ms] (mean, across all concurrent requests)  
Transfer rate: 244.49 [Kbytes/sec] received  
  
Connection Times (ms)  
 min mean[+/-sd] median max  
Connect: 0 36 131.8 0 519  
Processing: 17 2024 1101.8 2083 3654  
Waiting: 10 1979 1104.9 2072 3632  
Total: 17 2060 1094.7 2084 3654  
  
Percentage of the requests served within a certain time (ms)  
 50% 2084  
 66% 2602  
 75% 3121  
 80% 3121  
 90% 3644  
 95% 3644  
 98% 3644  
 99% 3654  
 100% 3654 (longest request)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Labels | -c 1 | -c 5 | -c 10 | -c 50 | -c 100 |
| Time taken for tests(seconds) | 0.734 | 0.324 | 0.318 | 3.653 | 3.663 |
| Total Transferred | 917000 | 917000 | 917000 | 917000 | 917000 |
| HTML transferred | 907900 | 907900 | 907900 | 907900 | 907900 |
| Time per Request | 7.337 | 3.243 | 3.176 | 36.527 | 36.628 |
| Requests per second | 136.29 | 308.35 | 314.84 | 27.38 | 27.30 |
| Transfer Rate | 1220.49 | 2761.25 | 2819.40 | 245.16 | 244.49 |

*\* Now, we have a cached proxy. When client send a request, proxy redirect that request to server then, store the responsed file. After that time, when a client wants same file then, it will be served from proxy not server. Therefore, server dont used in such kind of situation. Besides, we do not have to create a file, this decreases response times.*

We can see that values in the table are better than values in http server table. As we mentioned above files are served from proxy. Again, transfer rates are decresing when concurrency is too much. Reason of that, our proxy can not handle all of them in short times.