University of Tartu

–Institute of Computer Science–

Task 1: System Architectures and Performance Modelling

Distributed Systems - LTAT.06.007

Bruno Rucy
Mykyta Baliesnyi
Thamara Luup

Professor	:	Huber Flores
Teaching Assistants	:	Mohan Liyanage
		Farooq Dar

1 EXERCISE

1.1 SOLUTION I: SEPARATE SERVER FOR AUTHENTICATION

A separate server for authentication will be added. Every time a user connects to the server, the user will be directed to the authentication server, who will send a token to the client. When the user sends a request to the system, the token will be sent together with the request, thus the main server can process it and send back an answer. When the authentication is not required anymore, the authentication server can be suppressed and the main server can handle the requests normally. This solution is illustrated in Figure 1.1.

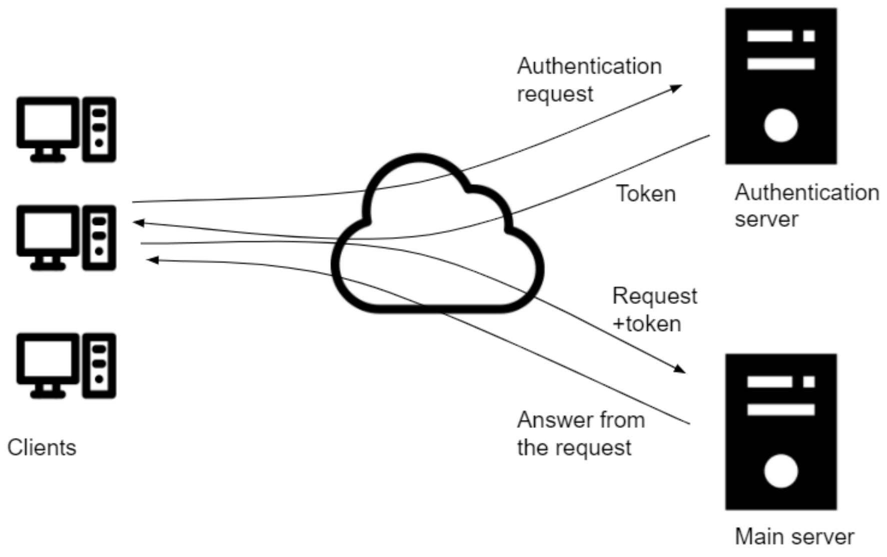

Figure 1.1: System with a separate server for authentication

1.2 USE OF INTERCEPTOR

The interceptor will break the usual flow of control and allow the authentication code to be executed. The server *per se* does not require authentication. When the authentication (implemented as an interceptor) is added or removed, there is no need to shut down the system and the modifications can be done on-the-fly. Figure 1.2 shows a call that is intercept to authenticate the client before the request is sent to the server.

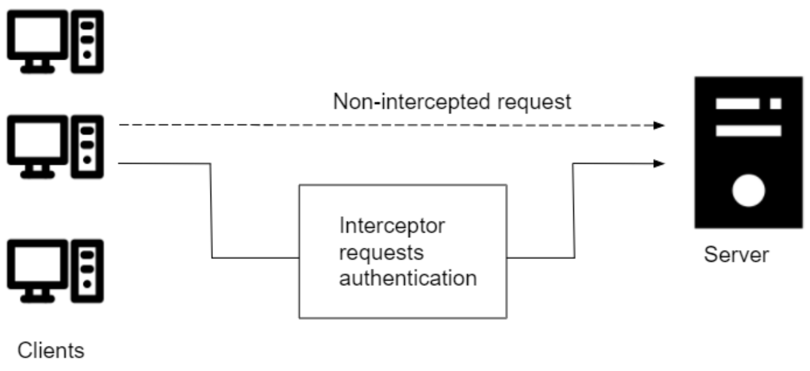

Figure 1.2: Interceptor handles authentication

2 EXERCISE

2.1 RESPONSE TIME

From the utilization law, the throughput of the system can be derived:

$$U_i = S_i \times X_i \tag{2.1}$$

$$X_i = \frac{U_i}{S_i} = \frac{0.60}{30 \cdot 10^{-3}} = 20tps \tag{2.2}$$

From the forced flow law:

$$X_0 = \frac{X_i}{V_i} = \frac{20}{4} = 5tps \tag{2.3}$$

Finally, the response time is calculated with the interactive response time law:

$$R = \frac{M}{X_0} - Z = \frac{100}{5} - 3 = 17s \tag{2.4}$$

2.2 IMPLICATIONS

In his book, Jakob Nielsen [2] discuss the response time for web applications. Three limits are mentioned:

- 0.1 second: that's the limit time for users think that the systems is reacting instantaneously;
- 1 second: the user will feel a small delay, but the user's flow of thought is kept;
- 10 seconds: that is the limit to keep user's attention and it is highly recommended to provide the user with a prediction of the time to complete the task.

The response time of the system is higher than 10 seconds, thus the user will loose the attention and will feel that the system is sluggish. In order to soften this issue, there should be an indication that the computer is working on the task and, if possible, an indication of the status, *e.g.* percentage of the task that is completed or estimation of the remaining time. If more resources are available, the system would benefit from identifying the bottleneck and replacing the component by one with superior performance.

3 EXERCISE

3.1 QN MODEL

The system was modelled and it is shown in Figure 3.1.

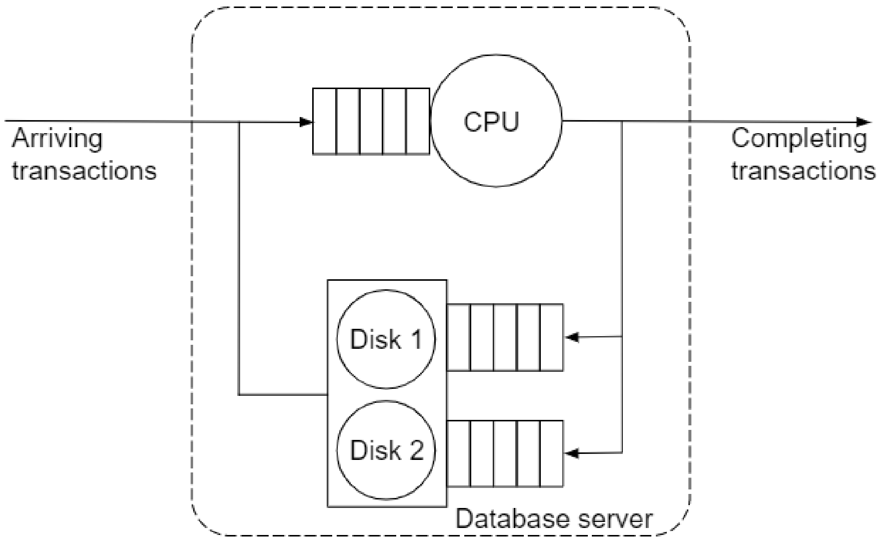

Figure 3.1: QN model for the web server

3.2 SERVICE DEMAND

For a resource i, the service demand D_i is defined by:

$$D_i = \frac{U_i \cdot T}{C_0} \tag{3.1}$$

Let D_{CPU}, D_{Disk1} and D_{Disk2} denote, respectively, the service demand for the CPU, Disk 1 and Disk 2. They are calculate with the previous equation and the result is as follows:

$$D_{CPU} = \frac{U_{CPU} \cdot T}{C_0} = \frac{0.25 \cdot 3,600}{141,600} = 0.00635s = 6.35ms/transaction \tag{3.2}$$

$$D_{Disk1} = \frac{U_{Disk1} \cdot T}{C_0} = \frac{0.35 \cdot 3,600}{141,600} = 0.00889s = 8.89ms/transaction \tag{3.3}$$

$$D_{Disk2} = \frac{U_{Disk2} \cdot T}{C_0} = \frac{0.30 \cdot 3,600}{141,600} = 0.00762s = 7.62ms/transaction \tag{3.4}$$

3.3 THROUGHPUT

The throughput is defined as the minimum between the server capacity and the offered workload. Here, it will be assumed that all $141,600$ requests are completely finished and there was no request that was not processed. Therefore the capacity of the server is equal or superior to the needs to complete the workload. This assumption is aligned to the fact that the utilisation of the resources is far from 100%. Although there is no information about the capacity of the server, the current throughput is calculated with the information about the workload and the previously made assumptions.

$$X_0 = \frac{C_0}{T} = \frac{141,600}{3,600} = 39.33tps \tag{3.5}$$

In a hypothetical scenario, the offered workload is high enough such that the utilisation of the system is at its limit. Because Disk 1 has the highest utilisation, it is assumed to be the bottleneck of the system when the workload increases and the utilisation of the system reaches its maximum. Therefore, the maximum throughput will happen when this resource is saturated, *i.e.*, when Disk 1 reaches the maximum utilisation. Thus, the maximum throughput in this hypothetical scenario is:

$$X_{max} = X_0 \cdot \frac{1}{U_{Disk1}} = 39.33 \cdot \frac{1}{0.35} = 112.37tps \tag{3.6}$$

3.4 RESPONSE TIME

The response time is given by:

$$R = \frac{S}{1 - \lambda S} \tag{3.7}$$

Because the system is not at maximum utilisation, it is reasonable to assume that all requests submitted are being completed, thus $A_0 = C_0$. Thus, the arrival rate λ becomes:

$$\lambda = \frac{A_0}{T} = \frac{C_0}{T} \tag{3.8}$$

Given that $B_i = U_i \cdot T$, the mean service time at resource i is:

$$S_i = \frac{B_i}{C_i} = \frac{U_i \cdot T}{C_i} \tag{3.9}$$

Assuming that all components of the system have a participation in every request, then $C_i = C_0$ and the mean service time becomes:

$$S_i = \frac{U_i \cdot T}{C_0} \tag{3.10}$$

Using Equations 3.8 and 3.10 in Equation 3.7 and rearranging:

$$R = \frac{\frac{U_i \cdot T}{C_0}}{1 - \frac{C_0}{T} \cdot \frac{U_i \cdot T}{C_0}} \tag{3.11}$$

$$R = \frac{U_i \cdot T}{C_0} \cdot \frac{1}{1 - U_i} \tag{3.12}$$

The first part of Equation 3.12 is the definition of the service demand, therefore:

$$R = \frac{D_i}{1 - U_i} \tag{3.13}$$

The total response time R is calculated by the sum of individual response times:

$$R = R_{CPU} + R_{Disk1} + R_{Disk2} = \frac{6.35}{1 - 0.25} + \frac{8.89}{1 - 0.35} + \frac{7.62}{1 - 0.30} = 33ms \tag{3.14}$$

3.5 RECOMMENDATIONS

According to [2], the user has have the feeling that the system is reacting instantaneously and the user's flow of thought is assured. Google PageSpeed Insights [1] recommends keeping the server response time under $200ms$. The utilisation of the resources are way below the limit, therefore the system can handle an increase in the service demand. Although the system is not working on the full capacity, adding redundant elements would allow the system to operate in case of failure of one component. It would also the requests to be processed faster in case the system stays idle for some time and then there are too many requests arriving at the same time.

4 EXERCISE

4.1 STATEFUL SERVER

In the other hand, a stateful server stores the status information of the clients and the information is not automatically deleted. All the data must be recovered in case of a server crash. Differently from a stateless server, it is possible to do pre-fetching. The stateful server can have very good performance without causing issues related to scalability or reliability.

Figure 4.1 shows an example of a stateful server. First, the Client requests to be logged in. The task is sent to Server 1, who will query the database and confirm the login. In a session variable, the server will save the state of the client, which is `Logged=true`. The client is not authenticated in all machines. If the client sends another Request to Server 1, the task will be performed, otherwise, it will be rejected.

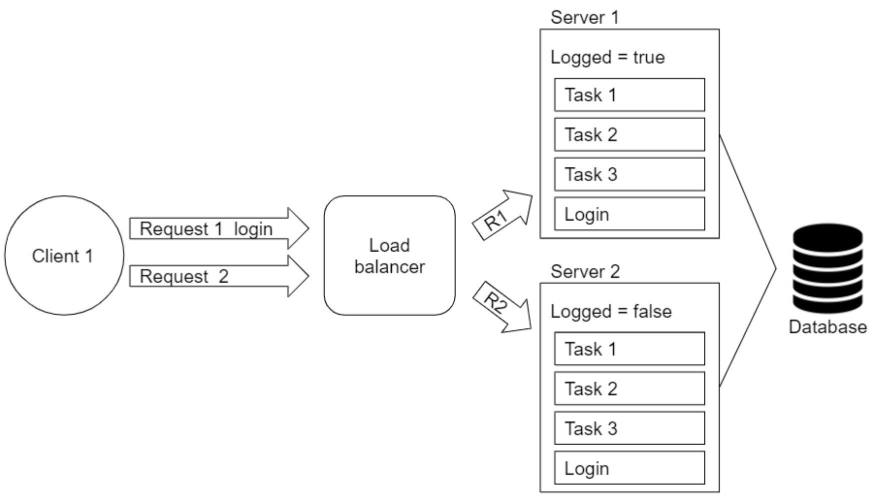

Figure 4.1: Stateful server

4.2 STATELESS SERVER

The main difference concerns the storage of the status of the clients. A stateless server does not retain information about the state of the clients. In some cases, some information can be kept, but the loss of the data does not interrupt the server's service [3]. There is no guarantee that the client's cache will be invalidated. The clients and the servers are autonomous and the crash of a client or server does not lead to conflicts. Additionally, the performance is eventually degraded.

An example of stateless server is shown in Figure 4.2. After the login request is sent by the Client 1, a token is sent back to the client. Afterwards, the client will send the server this token together with the new request. The token will be authenticated, independently of the server that is being used.

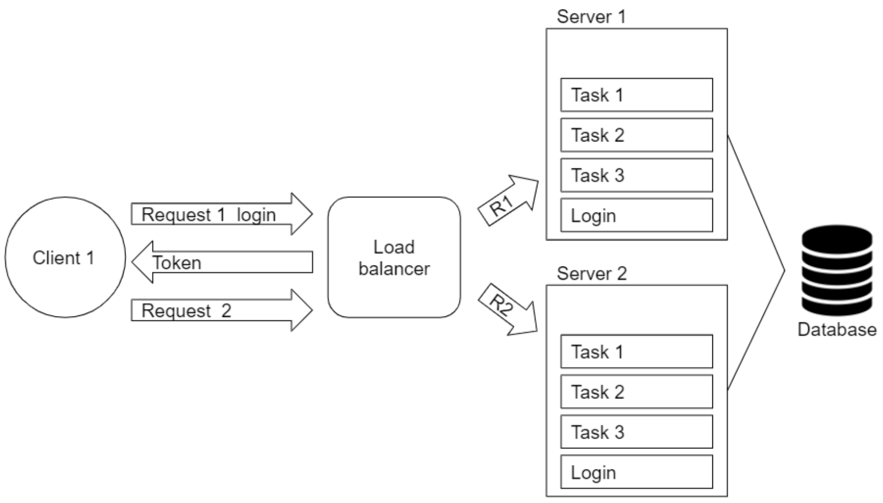

Figure 4.2: Stateless server

5 EXERCISE

The total duration that the service was down t_{down} is the sum of individual down times:

$$t_{down} = 10 + 3 + 5 + 5 + 7 + 1 = 31\, min \tag{5.1}$$

The three days is the total time t_{total}. The duration that the server was working t_{up} is given by:

$$t_{up} = t_{total} - t_{down} = 3 \times 24 \times 60 - 31 = 4289\, min \tag{5.2}$$

The availability of the site is the fraction that it was up, thus:

$$Availability = \frac{t_{up}}{t_{total}} = \frac{4289}{4320} = 0.9928 = 99.28\% \tag{5.3}$$

6 EXERCISE

6.1 DISK 1 REPLACEMENT

If Disk 1 is replaced by a disk that is 50% slower, the other resources are not affected. For the disk replaced, $(100\% + 50\%)$ of the original service demand will be needed. Thus, the new demands can be obtained by multiplying the old ones by the factor 1.5. Table 6.1 shows the new service demands.

	R1	R2	R3	R4
CPU	0.20	0.20	0.45	0.7
Disk 1	0.375	0.375	0.525	0.45
Disk 2	0.15	0.30	0.25	0.2
Disk 3	0.05	0.15	0.40	0.1

Table 6.1: Service demand in seconds, after Disk 1 is replaced

The overall service demand in the server of Disk 1 was 75 milliseconds before the replacement. Now, it will be $112.5ms$ and the service demand on the other disks are not altered.

6.2 REPLACEMENT OF DISKS 1, 2 AND 3

If the disks are replaced by disks that are 50% faster, $(100\% - 50\%)$ of the original service demand will be needed. Table 6.2 shows the new service demands, obtained by multiplication of old service demands by 0.5.

	R1	R2	R3	R4
CPU	0.20	0.20	0.45	0.7
Disk 1	0.125	0.125	0.175	0.15
Disk 2	0.075	0.15	0.125	0.1
Disk 3	0.025	0.075	0.20	0.05

Table 6.2: Service demand in seconds, after replacement of Disks 1, 2 and 3

The overall service demand in the server of Disk 1, 2 and 3 were, respectively, 75 milliseconds, 120 milliseconds and 140 milliseconds. Now, they assume values of $37.5ms$, $60ms$ and $70ms$, respectively.

6.3 RESPONSE TIME WITH INCREASED ARRIVAL RATE OF R1

The utilisation of a resource i for a request j is:

$$U_{ij} = D_{ij} \cdot \lambda_j \tag{6.1}$$

The total utilisation of a resource i is the sum of the utilisation for N requests:

$$U_i = \sum_{j=1}^{N} D_{ij} \cdot \lambda_j \tag{6.2}$$

The values of utilisation of each resource and per type of request are shown in the following table:

Resource	R1	R2	R3	R4	Total
CPU	0.03	0.05	0.18	0.08	0.344
Disk 1	0.04	0.06	0.14	0.04	0.276
Disk 2	0.02	0.08	0.10	0.02	0.222
Disk 3	0.01	0.04	0.16	0.01	0.217

Table 6.3: Utilisation per type of request

The total response time for a request j is the sum of the individual response times of the M resources:

$$R_j = \sum_{i=1}^{M} R_i \tag{6.3}$$

Table 6.4 contains the response time per resource, obtained from Equations 3.13 and 6.3 and Table 6.3.

Resource	R1	R2	R3	R4
CPU	0.305	0.305	0.686	1.067
Disk 1	0.345	0.345	0.483	0.414
Disk 2	0.193	0.385	0.321	0.257
Disk 3	0.064	0.192	0.511	0.128
Total	0.907	1.227	2.001	1.886

Table 6.4: Response time per type of request

When the arrival rate λ_1 increases, the values of utilisation are recalculated considering $\lambda_1 = 1.7 \cdot 0.15 = 0.255$. Table 6.4 is updated and becomes Table 6.5. The response time of request R1 is increased from 0.907 to 0.934, which corresponds to approximately 3%.

Resource	R1	R2	R3	R4
CPU	0.315	0.315	0.709	1.102
Disk 1	0.358	0.358	0.502	0.430
Disk 2	0.197	0.193	0.514	0.129
Disk 3	0.064	0.193	0.514	0.129
Total	0.934	1.259	2.052	1.923

Table 6.5: Response time for $\lambda_1 = 0.255$

6.4 RESPONSE TIME WITH INCREASE ARRIVAL RATE OF R3

When the arrival rate λ_3 increases, the values of utilisation are recalculated considering $\lambda_3 = 1.45 \cdot 0.40 = 0.58$. Table 6.4 is updated and becomes Table 6.6. The total response time of request R3 increased from 2.001 to 2.216, which is approximately 10.7%.

Resource	R1	R2	R3	R4
CPU	0.348	0.348	0.783	1.217
Disk 1	0.378	0.378	0.530	0.454
Disk 2	0.204	0.409	0.341	0.273
Disk 3	0.070	0.211	0.563	0.141
Total	1.001	1.346	2.216	2.085

Table 6.6: Response time for $\lambda_3 = 0.58$

6.5 RESPONSE TIME WHEN R2 ARE RUN ON A DIFFERENT COMPUTER

When the arrival rate λ_2 increases, the values of utilisation are recalculated considering $\lambda_2 = 0 \cdot 0.25 = 0.0$. Table 6.4 is updated and becomes Table 6.7. The total response time of request R2 decreased from 0.907 to 0.838, which corresponds to approximately -7.6%.

Resource	R1	R2	R3	R4
CPU	0.283	0.283	0.637	0.992
Disk 1	0.318	0.318	0.445	0.381
Disk 2	0.176	0.351	0.293	0.234
Disk 3	0.061	0.183	0.488	0.122
Total	0.838	1.135	1.863	1.729

Table 6.7: Response time for $\lambda_2 = 0$

BIBLIOGRAPHY

[1] *Improve Server Response Time - PageSpeed Insights - Google Developers*. URL: `https://developers.google.com/speed/docs/insights/Server`.

[2] Jakob Nielsen. *Usability Engineering*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1993. ISBN: 0125184050.

[3] Andrew S Tanenbaum and Maarten Van Steen. *Distributed Systems; 3rd ed.* Upper Saddle River, NJ: Prentice-Hall, 2008.