Are you Load Balancing wrong?

Egger Daniel
1518462

Steinmüller Philipp
1518369

June 1, 2017

Contents

Abstract

A short paper about load balancers and problems that come with such a system.

1 Introduction

In this paper we are talking about load balancing. Inspired by "Are You Load Balancing Wrong?" [3] we are giving a short summary of what load balancing is and talk about some problems that might occur when using load balancers. Later we will discuss the most common areas in which load balancers are deployed and a brief overview of the maintainability of load balancing systems. In addition we also discuss why benchmarking is important for load balancing. At the end of the document we summarise some aspects of future development of load balancers.

2 What is Load Balancing?

In this section we will explain the basics of the concept of load balancing based on web-servers, since this is a common area of deployment for such systems. The content of this explanation is based on "Load Balancing in the Context of Virtualization" [2].

2.1 What is Load Balance used for?

If a company would host their website on a single server there is a high possibility that this server will eventually stop working. This can either be because the requests that the server receives are too much for it or simply because the physical pieces of the server stop working.
To avoid such problems the concept of load balance is introduced. Instead of one expensive large server, companies use server clusters with many, not so expensive, servers. An example for this is shown in Figure 1. In these clusters load balancers are used to distribute the incoming requests to different servers to avoid overloading a single one or giving the request to a broken one.
This also results in a higher resiliency. Instead of a disconnection when a server breaks down, the service will continue but the load balancer will pick a different one to guarantee a smooth continuation of the service.
So two of the main reasons to use a load balancer are:

- Increased capacity to make service faster
- Resiliency to be more secured against system failure

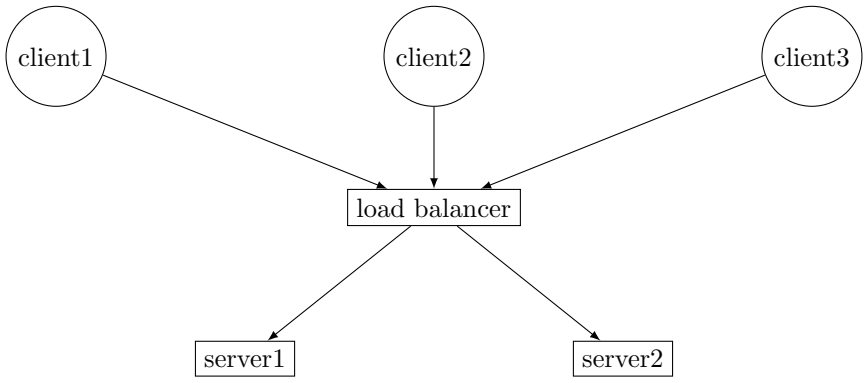

Figure 1: simple visualisation of a load balancer

2.2 Example types of Load Balancers

In this subsection we will give a brief overview of two commonly used load balancing types for web-servers. While these and other methods might not be a suitable solution on their own, a combination can offer a system to load balance in server clusters.

2.2.1 DNS-Method

The DNS-Method is based on "Domain Name Servers'. Therefore a company need multiple IP-addresses that are accessible with the same domain.
With this method the DNS decides, which server gets which request. This procedure can be random or follow a fixed pattern. This form of load balancing is also called "Round-Robin DNS".
Although this is a simple way to implement a simple concept of load balancing it does not fulfil the real purpose of such a system, since it does not know which server is close to its limit, since with this method many computation-heavy requests could reach one server while another one is running idle.

2.2.2 NAT-Method

With this method a real load balancer is introduced.
Instead of multiple IP-addresses a single one is sufficient for this method, since the forwarding will be handled inside a subnet via a load balancer. The servers have to be synchronised when this method is applied because the information stored should be accessible from every one of them.
With this method it is possible to create a ranking of the current load on each server. So the load balancer can forward the incoming request to the server that is currently most suitable for the task.

3 Field of Application

This section describes where load balancing is used. The purpose of use, as well as the functionality, is explained in more detail with a brief example.

3.1 Increased Capacity

Load balancers are widely used by many web servers. The reason for this is that, with a large number of clients, many requests are generated and these may overload a single server machine. By using a load balancer, the requests can be distributed to several server machines. This increases the number of possible requests a server can handle.
For example, if a server machine can answer 100 queries per second, it is possible to process 200 queries per second by adding a second server machine. Thus, the server is scalable.

3.2 Resiliency

In addition to increasing the maximum capacity, load balancers are also used as a means against failures of a server machine.
As an example, four server machines are considered, each of which can process 100 queries per second. If 300 requests are received per second, each machine processes 75 requests per second. When a machine fails, the load balancer ensures that the requests to the failed machine are routed to the three functioning machine. These are then obtained in about 100 queries per second, which corresponds to their maximum capacity. However, it is ensured that the system continues to run smoothly if a machine fails.

4 Maintainability

The use of a load balancer not only offers the advantages described in Field of Application (3), but also facilitates the maintenance of the server system. If a server computer is currently in a maintenance state because configurations are being processed or updates are provided, this is detected by the load balancer and the requests to the computer are routed to other server computers [1]. As a result, there is no perceptible loss of performance with a user.

5 Problems of Load Balancing

While load balancers offer a lot of advantages there are also some problems that might occur if you not plan the implementation of a load balancer beforehand, especially when working in a team.

5.1 Capacity

You and your team have to plan ahead and compromise on a way to use load balancers. You either use them to offer a higher capacity for faster access or to make your system more reliable in case of a server failure.
Since both ways use a different approach you have to determine the way you use capacity beforehand. The Problem is that you need to know what capacity will be needed to handle all the requests. This is hard to determine but there are some benchmarks (6) to approximate the rough capacity that will be needed.

5.2 Miscommunication

Your team needs to know which way you are taking so there wont be situations that one member optimises the system to offer higher capacity while another is working on a more resiliency based system.

5.3 Monitoring

When the system is deployed it needs to be monitored. The system needs to send an alarm message if there is an problem. This will need to be acknowledged x minutes before a system failure. x, in this situation, is the amount of time it takes to replace one server in the system so there will be no time, when the total capacity of the load balance system will be lower than planned.

6 Benchmarks

6.1 Why use Benchmarks?

In order to be able to assess how many requests each server can process per second, it is not sufficient to look at only the source code. The code does not indicate how much time or how many resources are used per request. Even if one were to know the theoretically required time and resources per query, this would not be sufficient. In order to be able to determine the capacity of a server computer, benchmarks are performed.

6.2 Determining the number of possible Requests

With these benchmarks, queries are generated and sent to the server. The response times of the server are recorded. Assuming a response time of 100ms is sufficient, start with e.g. 25 requests per second. The number of these requests is now increased until the response time of the server is more than 100ms. This procedure can be used to determine how many requests per second can be handled by the server.

6.3 Queries

However, the problem here is that not all queries require the same time or resources. A pure read-only request is executed much faster than a read-write request. This fact must be taken into account so as not to distort the benchmark. Most sites use a percentage method. In this method for example, 90% of the requests to the server must have a response time less than 100ms. This procedure eliminates the few queries, which require a lot of time and resources, and does not distort the benchmark. If, in comparison to the percentage method, the average value of the response times is used, the result would be impaired and corrupted by a few very long response times.
Another point in terms of the benchmarks are the queries themselves. As described above, different queries can be answered differently fast. For example, the test should ensure that both short and long requests are tested on the server. A server may be able to answer 120 short queries per second, but only 80 long queries per second, which are requiring more time and resources.

6.4 When to benchmark?

Furthermore, such benchmarks should be made for each new release, since even small changes can have an effect on the processing time. It is problematic if the benchmarks are only occasionally executed. If it happens that there is a negative performance change after a few releases, the source can only be found with very much effort. If the benchmark system is automated, which means that before every release a benchmark is performed automatically, it is very easy to find the source for the negative performance change.

7 Future of Load Balancing

It is difficult to make assumptions about the future of load balancing. For load balancing to work well, its hardware and software must handle a large amount of information. If the demands on servers continue to expand in the future, it is conceivable that load balancing itself leads to a weak point in the system, since this could be a bottleneck. To prevent this from happening, the systems must have enough computing power and also require more resources.

8 Summary

Load balancing is a very important way to utilise the full potential of a network of machines. Although it offers many advantages, it also comes with some problems. But by informing yourself beforehand, a good communication with your team and calculation of capacity based on benchmarks, these problems can be eliminated or at least be suppressed enough to implement a stable load balancing system.

References

[1] 1&1. Bessere serverzugriffszeiten durch load balancer, 2016.

[2] Stepniowski Daniel. Load balancing in the context of virtualization. 2009.

[3] Thomas A. Limoncelli. Are you load balancing wrong? *Commun. ACM*, 60(2):55–57, January 2017.