medium.com

Clustering vs Distributed Systems - The Startup - Medium

Lakshitha Samarasingha

7-9 minutes

This post contains high-level concepts on clustering load balancing and distributed systems. It mainly focuses on the aspects of what, why, advantages and disadvantages.

What is clustering?

When it comes to a model like a client and a server application model, the client sends requests to the server in order to execute a task and get results. This server can be a single endpoint that contains a single server entity or in other words, it can be one service entity. Let’s say a client sends a request to this single service endpoint which contains multiple sub-tasks. Then once it gets the request it starts processing the sub-tasks in order to complete the request but it has to perform all the sub-tasks by using its own resources. So it will take some time to execute all the sub-tasks to complete the request.

So with this approach imagine what happens if this server (the single service endpoint) has thousands of clients and at the same time all the clients start to communicate with the server by sending requests. The server may run out of its resources during the execution of the tasks, so some clients will have to wait until the server gets its resources to fulfill the needs of the clients. The availability of resources becomes low and it adds limits to the scalability as well. The end result would be a huge decrease in performance.

What happens if the server goes down? it will make services unavailable to the clients and till the server comes up clients will have to hold their tasks. This may introduce problems to the application users and it may cause data losses as well.

Now let’s see how to overcome the above problems. In order to overcome the resource problem and increase performance, we can add more resources to the same endpoint to execute sub-tasks parallelly. In other words, we can add more service entities that can communicate with each other and execute sub-tasks parallelly. All these service entities or server entities will share one service endpoint where all clients would connect and send requests. So if the number of requests or number of connected clients or users are getting high we can add more service entities to the same endpoint. This concept is called “**scalability**” in clustering.

Unlike the single service entity or single server now we have multiple service entities to fulfill the requests. Let’s assume one service entity goes down due to an error occurred, now the requests coming to that server entity will be redirected to the other available service entities by minimizing the downtime of the services. With this ability, the users or clients may not experience huge availability of the services since there are other service entities that can take over the execution by minimizing the downtime. This concept is known as “**high availability**”

Now let’s take a look at the summary of the things that we discussed above

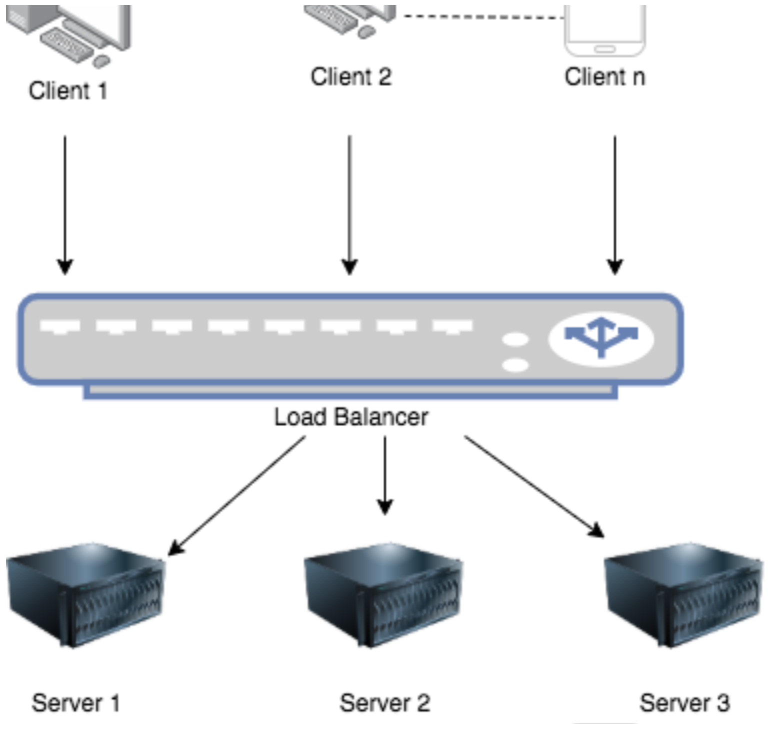
Clustering in a nutshell

Clustering is the concept of multiple service nodes or entities working as one single entity by sharing a single endpoint to enable the scalability and high availability of the services.

As explained above ,the main advantages of clustering **scalability, high availability, increased performance, and simplicity**.

Now we get another problem in our minds, how to handle and route requests which come to a single endpoint and distribute those requests between service entities or nodes in a cluster?

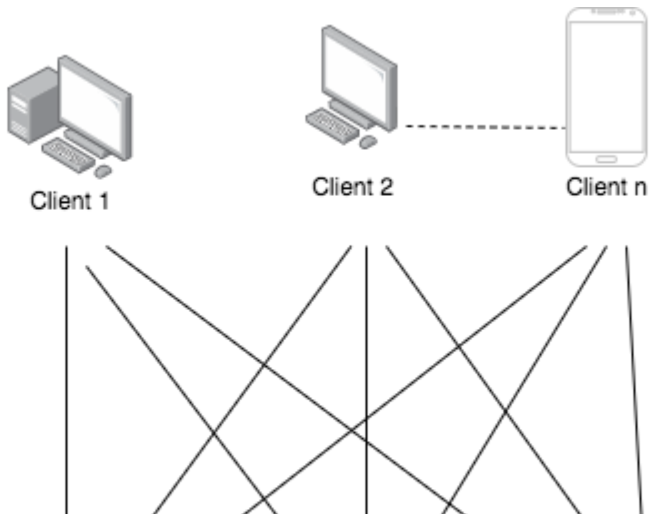
The answer is we have to use a load balancer to handle the requests which come to the endpoint and distribute them between service nodes in a cluster using a proper mechanism.
(I will be explaining more about load balancing in a different blog post in the future.)

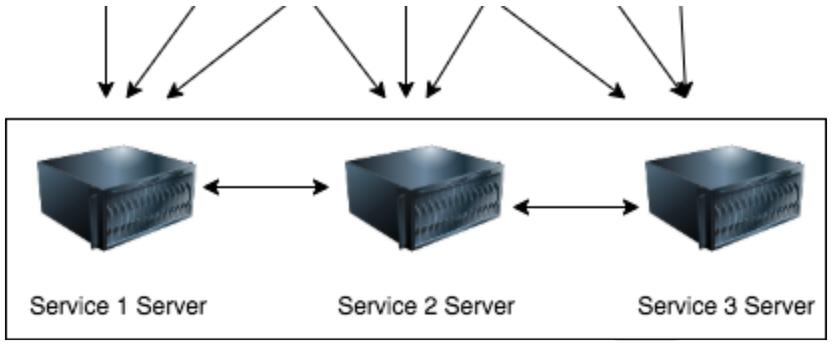


Cluster using a load balancer

What are distributed systems?

Now let's assume that there is a system that can interpret an incoming request and divide the request into sub-tasks and re-route those tasks into different endpoints (in other words different service modules) that are dedicated to complete those tasks. This concept of having dedicated servers or endpoints for different tasks or service modules is called a distributed system. Here the functionalities are being distributed among different dedicated servers.





distributed system

This adds few advantages like increasing the ability of concurrent access to a particular service, reduce the resource consumption, reusability of common services between applications and it will also make it easy to extend the functionalities by adding them in a distributed way.

What is the difference between a distributed system and a cluster?

Mainly there are two major differences between a distributed system and a cluster. The first major difference comes with the way the services are being executed. In distributed systems, it distributes sub-tasks in different servers, so if there are main 10 sub-tasks then there will be 10 endpoints (or in other words 10 different dedicated servers) to get the work done but in clustering, one endpoint is there and that endpoint is being mapped to several servers (through a load balancer) each consists of the ability to execute 10 sub-tasks within itself. Now given the above fact think what happens if one server fails in a distributed system? Then the whole system would have to wait till that server comes up to complete the task or to abort the task. But what happens if one server fails in a clustering system? The load balancer would redirect the incoming requests to another service entity or server node to handle the task

Let's take another example, assume that there is an application that has thousands of clients sending multiple requests at a time, so in order to decrease the execution time of a request we can distribute the functionalities as sub-tasks in a distributed system. Due to the dedication and high resource availability, it will decrease the execution time of a single sub-task and will increase efficiency. When it comes to a cluster it will increase the number of tasks that are being executed during a given time period and will increase the efficiency through that, and that is the second major difference.

In simple words assume that a task consists with 10 sub-tasks and one sub-task takes 1 hour to complete, then for a single server to complete the task it will take 10 hours.

Now assign that task to a distributed system where we have those 10 sub-tasks distributed in 10 different dedicated servers, with parallel execution it will complete all the sub-tasks within 1 hour.

In a clustering system with a single server, it will still take 10 hours to complete the whole task but we can increase the number of servers in clustering and let's say we increased it up to 10 servers and now at a given time the cluster is now capable of handling 10 requests parallelly.

Now comes the main question which is…

When to use which strategy?

Well, I would say it is highly depending on the properties of your application like the number of users per given time, number of requests per given time, number of database connections at a given time and load of the data that needs to handle etc. likewise there are many things you have to consider when selecting what approach is more suitable for your application.
But most of the time if you are having a large scale application then having a combination of both would give you more results, as an example, you can use clustering inside of a distributed system by

clustering the dedicated servers for more availability and scalability but still, everything depends on your application’s properties.