Implementation Of A Scalable Architecture

David Alejandro Vasquez Carreño
Ingeniería de sistemas
Escuela Colombiana de Ingeniería Julio Garavito
Bogotá, Colombia
david.vasquez@mail.escuelaing.edu.co

***Abstract*—In today's world, most companies support their products ans services in IT systems, like web servers, bank functions or cloud gaming, but there is a more complex architecture behind this services, that allow companies to provide continuous and scalable architectures to the growing set of customers.**

***Index Terms*—Server, CPU, Load.**

I. Introduction

Complex systems have mechanisms for allowing more users, for be able to handle all requests for the company, In data centers, there is scalable mechanisms like load balancers of backup servers that are there for failings in the system. This allow the company to support services 24/7 without major interruptions, except for maintenance of servers, that is previously negotiated with the provider.

Concurrent request form users are handled by a capable data center, that provide multiple servers in a paid plan, but this systems are often limited or can be sensible to a excessive number of requests, that are not specially planned by the company, like special offers in black Friday, that can take web sites to have more users than expected, taking servers to the limit, in networking and CPU load. Data centers try to make the server tasks easier, by distributing the requests between more servers that are not handling that load, so the users can't notice so many concurrence in the system.

Another mechanism, that is the one we are going to be discussing, is the one to launch more servers depending on the CPU load or network load of a system, so more servers are available in case of the increment of users requests, like in our example of black Friday customers going to buy at the same time at a digital store.

II. What is an scalable architecture?

Scalability is a property of the systems to adapt itself and react without losing quality. In our case of study, we can look at network, processes or systems, that must change based on the environment and parameters, and make changes to maintain quality, [1]

The image[1] shows an example of horizontal scaling, that consists in adding more servers to the layout, keeping the properties of those, to be equal to the already existing ones.

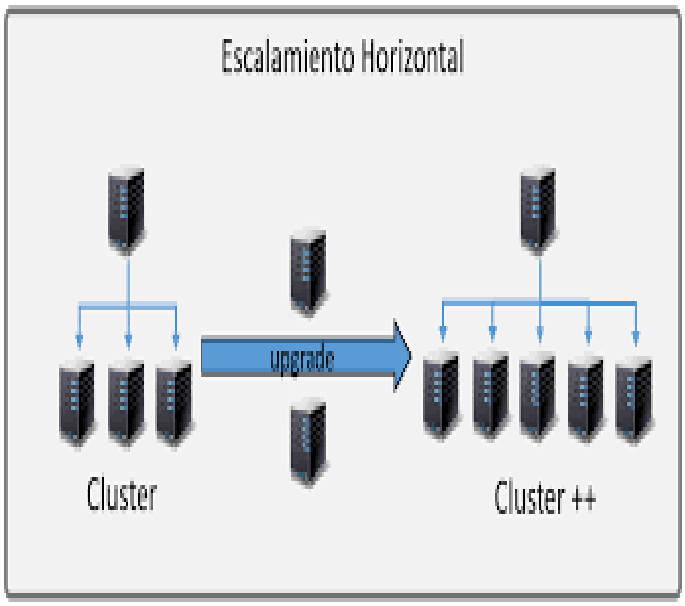

Fig. 1. Horizontal scaling

Another example of scaling, is vertical scaling, that allow us to equally add more servers, but in this case, the servers are better, have better processing capabilities or networking interfaces, etc.

III. Load criteria

For being able to determine whether a server has a high load, we must choose a criteria for making that decision. Some examples of server load are the following:

- CPU load - Most used
- Memory usage
- Disk usage
- Memory usage
- Network usage
- GPU load - Rare

The figure 2 shows an example of a linux tool for checking system properties like CPU, memory and other characteristics of a linux machine, that is the most common thing nowadays in servers. For monitoring these properties, more complex

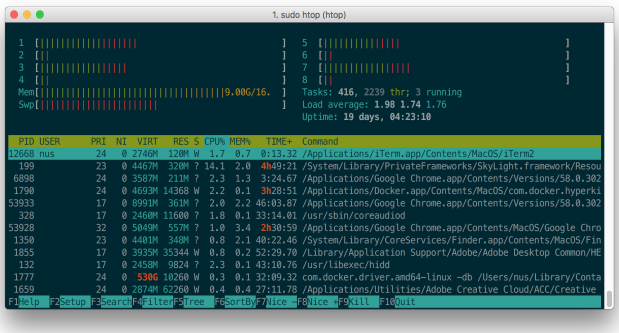

Fig. 2. Linux top

and complete monitoring programs are used, like nagios or windows based monitors. [2]

IV. Solution

Our implementation is the simple creation of server based on the CPU load of a single machine, that we are going to call ”FIRST SERVER”, and it is going to be monitored via Amazon Web Services, that provide us EC2 machines to work and make tests.

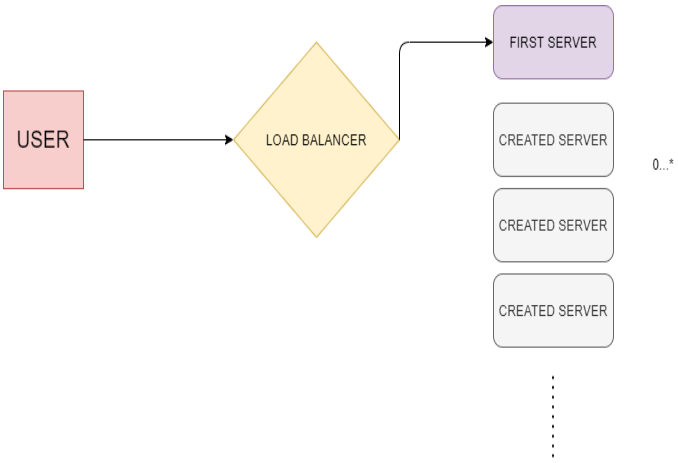

Fig. 3. Design

A java spark server receives Http request for the calculation of prime numbers in a big range, and with a given number of threads, that are made for increase the CPU load, and force the creation of more identical machines. The application is a java dockerized application that receives post request to ”primes?inf=numberAsup=numberBthreads=number”.

V. Evaluation

With our simple implementation, we were able to see how AWS creates new servers based on a given criteria, like CPU load or network packages. This implementation is not meant to decrease the load in the first server, but for showing the creation of identical machines given a configured image of the first one.

A interesting case study would be to implement a load balance in the middle, to distribute the request between the servers. In this case, the processing requests are meant to be unique and very CPU heavy.

VI. Conclusions

Scalable architectures are key in today technologies, for being able to maintain existing services of important companies that keep their services in cloud or internal data centers.

In real life, multiple mechanisms are used for maintaining this services, and we must evaluate which are adequate for the needed task.

References

[1] J. Carrillo, “Arquitectura web escalable.” https://www.solvetic.com/tutoriales/article/1929-arquitectura-web-escalable/#:~:text=La%20escalabilidad%20es%20la%20propiedad,calidad%20en%20los%20servicios%20ofrecidos., September 2015. Accessed on 2020-11-02.

[2] R. Chesley, “How to check server load on a windows server.” https://www.liquidweb.com/kb/check-server-load-windows-server/, March 2019. Accessed on 2020-11-02.