Scaling techniques for cloud-based applications involve adjusting the resources and infrastructure to handle varying workloads efficiently. Here are some common scaling techniques:

1. Vertical Scaling (Scaling Up): It involves increasing the capacity of individual resources, such as upgrading to a more powerful server or adding more CPU, RAM, or storage to handle increased load. This approach can be limited by the maximum capacity of a single resource.
2. Horizontal Scaling (Scaling Out): It involves adding more instances of resources, such as adding more servers or virtual machines, to distribute the workload. This approach allows for better utilization of resources and provides the ability to handle larger loads.
3. Load Balancing: Load balancing distributes incoming traffic across multiple servers or instances to ensure that no single resource is overwhelmed. It helps in achieving better performance, high availability, and fault tolerance. Load balancers can be implemented at various layers, including the application layer, network layer, or DNS level.
4. Auto Scaling: Auto scaling is an automated approach to dynamically adjust resources based on demand. It involves monitoring application metrics, such as CPU utilization, network traffic, or request queue length, and automatically scaling resources up or down accordingly. It helps in maintaining optimal performance and cost-efficiency.
5. Containerization: Containerization platforms like Docker allow applications to be packaged along with their dependencies into lightweight, isolated containers. Containers can be easily replicated and deployed across multiple servers, enabling efficient horizontal scaling. Container orchestration tools like Kubernetes help manage and scale containerized applications effectively.
6. Serverless Computing: Serverless computing abstracts the underlying infrastructure and allows developers to focus solely on writing code. Scaling is handled automatically by the cloud provider, which provisions resources as needed based on incoming requests. It eliminates the need for manual scaling and provides a highly elastic and cost-efficient environment.
7. Caching: Caching involves storing frequently accessed data or computed results in memory or fast storage. By caching data, subsequent requests for the same data can be served quickly without the need for processing or retrieving it from the backend. Caching can significantly improve application performance and reduce the load on backend resources.
8. Database Scaling: Scaling databases can be challenging due to their inherent complexities. Techniques like database replication, sharding, or partitioning can be employed to distribute the database load across multiple instances or clusters. Additionally, adopting managed database services offered by cloud providers can simplify database scaling.
9. Content Delivery Networks (CDNs): CDNs cache and distribute content across a network of edge servers located in various geographic locations. By caching content closer to end users, CDNs reduce latency and improve performance. CDNs are particularly useful for applications that serve static content or have a global user base.

These scaling techniques can be used individually or in combination depending on the specific needs and architecture of the cloud-based application. It's important to analyze and understand the application's workload patterns, performance requirements, and cost considerations to determine the most suitable scaling strategy.

Load Balancing in Cloud Computing

Load balancing is the method that allows you to have a proper balance of the amount of work being done on different pieces of device or hardware equipment. Typically, what happens is that the load of the devices is balanced between different servers or between the CPU and hard drives in a single cloud server.

Load balancing was introduced for various reasons. One of them is to improve the speed and performance of each single device, and the other is to protect individual devices from hitting their limits by reducing their performance.

Cloud load balancing is defined as dividing workload and computing properties in cloud computing. It enables enterprises to manage workload demands or application demands by distributing resources among multiple computers, networks or servers. Cloud load balancing involves managing the movement of workload traffic and demands over the Internet.

Traffic on the Internet is growing rapidly, accounting for almost 100% of the current traffic annually. Therefore, the workload on the servers is increasing so rapidly, leading to overloading of the servers, mainly for the popular web servers. There are two primary solutions to overcome the problem of overloading on the server-

* First is a single-server solution in which the server is upgraded to a higher-performance server. However, the new server may also be overloaded soon, demanding another upgrade. Moreover, the upgrading process is arduous and expensive.
* The second is a multiple-server solution in which a scalable service system on a cluster of servers is built. That's why it is more cost-effective and more scalable to build a server cluster system for network services.

Cloud-based servers can achieve more precise scalability and availability by using farm server load balancing. Load balancing is beneficial with almost any type of service, such as HTTP, SMTP, DNS, FTP, and POP/IMAP.

It also increases reliability through redundancy. A dedicated hardware device or program provides the balancing service.

## **Different Types of Load Balancing Algorithms in Cloud Computing:**

### **1. Static Algorithm**

Static algorithms are built for systems with very little variation in load. The entire traffic is divided equally between the servers in the static algorithm. This algorithm requires in-depth knowledge of server resources for better performance of the processor, which is determined at the beginning of the implementation.

However, the decision of load shifting does not depend on the current state of the system. One of the major drawbacks of static load balancing algorithm is that load balancing tasks work only after they have been created. It could not be implemented on other devices for load balancing.

### **2. Dynamic Algorithm**

The dynamic algorithm first finds the lightest server in the entire network and gives it priority for load balancing. This requires real-time communication with the network which can help increase the system's traffic. Here, the current state of the system is used to control the load.

The characteristic of dynamic algorithms is to make load transfer decisions in the current system state. In this system, processes can move from a highly used machine to an underutilized machine in real time.

### **3. Round Robin Algorithm**

As the name suggests, round robin load balancing algorithm uses round-robin method to assign jobs. First, it randomly selects the first node and assigns tasks to other nodes in a round-robin manner. This is one of the easiest methods of load balancing.

Processors assign each process circularly without defining any priority. It gives fast response in case of uniform workload distribution among the processes. All processes have different loading times. Therefore, some nodes may be heavily loaded, while others may remain under-utilised.

### **4. Weighted Round Robin Load Balancing Algorithm**

Weighted Round Robin Load Balancing Algorithms have been developed to enhance the most challenging issues of Round Robin Algorithms. In this algorithm, there are a specified set of weights and functions, which are distributed according to the weight values.

Processors that have a higher capacity are given a higher value. Therefore, the highest loaded servers will get more tasks. When the full load level is reached, the servers will receive stable traffic.

### **5. Opportunistic Load Balancing Algorithm**

The opportunistic load balancing algorithm allows each node to be busy. It never considers the current workload of each system. Regardless of the current workload on each node, OLB distributes all unfinished tasks to these nodes.

The processing task will be executed slowly as an OLB, and it does not count the implementation time of the node, which causes some bottlenecks even when some nodes are free.

### **6. Minimum To Minimum Load Balancing Algorithm**

Under minimum to minimum load balancing algorithms, first of all, those tasks take minimum time to complete. Among them, the minimum value is selected among all the functions. According to that minimum time, the work on the machine is scheduled.

Other tasks are updated on the machine, and the task is removed from that list. This process will continue till the final assignment is given. This algorithm works best where many small tasks outweigh large tasks.

### **Load balancing solutions can be categorized into two types -**

* **Software-based load balancers:** Software-based load balancers run on standard hardware (desktop, PC) and standard operating systems.
* **Hardware-based load balancers:** Hardware-based load balancers are dedicated boxes that contain application-specific integrated circuits (ASICs) optimized for a particular use. ASICs allow network traffic to be promoted at high speeds and are often used for transport-level load balancing because hardware-based load balancing is faster than a software solution.

## **Major Examples of Load Balancers -**

* **Direct Routing Request Despatch Technique:** This method of request dispatch is similar to that implemented in IBM's NetDispatcher. A real server and load balancer share a virtual IP address. The load balancer takes an interface built with a virtual IP address that accepts request packets and routes the packets directly to the selected server.
* **Dispatcher-Based Load Balancing Cluster:** A dispatcher performs smart load balancing using server availability, workload, capacity and other user-defined parameters to regulate where TCP/IP requests are sent. The dispatcher module of a load balancer can split HTTP requests among different nodes in a cluster. The dispatcher divides the load among multiple servers in a cluster, so services from different nodes act like a virtual service on only one IP address; Consumers interconnect as if it were a single server, without knowledge of the back-end infrastructure.
* **Linux Virtual Load Balancer:** This is an open-source enhanced load balancing solution used to build highly scalable and highly available network services such as HTTP, POP3, FTP, SMTP, media and caching, and Voice over Internet Protocol (VoIP) is done. It is a simple and powerful product designed for load balancing and fail-over. The load balancer itself is the primary entry point to the server cluster system. It can execute Internet Protocol Virtual Server (IPVS), which implements transport-layer load balancing in the Linux kernel, also known as layer-4 switching.

## **Types of Load Balancing**

You will need to understand the different types of load balancing for your network. Server load balancing is for relational databases, global server load balancing is for troubleshooting in different geographic locations, and DNS load balancing ensures domain name functionality. Load balancing can also be based on cloud-based balancers.

### **Network Load Balancing**

Cloud load balancing takes advantage of network layer information and leaves it to decide where network traffic should be sent. This is accomplished through Layer 4 load balancing, which handles TCP/UDP traffic. It is the fastest local balancing solution, but it cannot balance the traffic distribution across servers.

### **HTTP(S) load balancing**

HTTP(s) load balancing is the oldest type of load balancing, and it relies on Layer 7. This means that load balancing operates in the layer of operations. It is the most flexible type of load balancing because it lets you make delivery decisions based on information retrieved from HTTP addresses.

### **Internal Load Balancing**

It is very similar to network load balancing, but is leveraged to balance the infrastructure internally.

Load balancers can be further divided into hardware, software and virtual load balancers.

### **Hardware Load Balancer**

It depends on the base and the physical hardware that distributes the network and application traffic. The device can handle a large traffic volume, but these come with a hefty price tag and have limited flexibility.

### **Software Load Balancer**

It can be an open source or commercial form and must be installed before it can be used. These are more economical than hardware solutions.

### **Virtual Load Balancer**

It differs from a software load balancer in that it deploys the software to the hardware load-balancing device on the virtual machine.

## **WHY CLOUD LOAD BALANCING IS IMPORTANT IN CLOUD COMPUTING?**

Here are some of the importance of load balancing in cloud computing.

### **Offers better performance**

The technology of load balancing is less expensive and also easy to implement. This allows companies to work on client applications much faster and deliver better results at a lower cost.

### **Helps Maintain Website Traffic**

Cloud load balancing can provide scalability to control website traffic. By using effective load balancers, it is possible to manage high-end traffic, which is achieved using network equipment and servers. E-commerce companies that need to deal with multiple visitors every second use cloud load balancing to manage and distribute workloads.

### **Can Handle Sudden Bursts in Traffic**

Load balancers can handle any sudden traffic bursts they receive at once. For example, in case of university results, the website may be closed due to too many requests. When one uses a load balancer, he does not need to worry about the traffic flow. Whatever the size of the traffic, load balancers will divide the entire load of the website equally across different servers and provide maximum results in minimum response time.

### **Greater Flexibility**

The main reason for using a load balancer is to protect the website from sudden crashes. When the workload is distributed among different network servers or units, if a single node fails, the load is transferred to another node. It offers flexibility, scalability and the ability to handle traffic better.

Because of these characteristics, load balancers are beneficial in cloud environments. This is to avoid heavy workload on a single server.

## **Conclusion**

Thousands of people have access to a website at a particular time. This makes it challenging for the application to manage the load coming from these requests at the same time. Sometimes this can lead to system failure.

https://www.javatpoint.com/load-balancing-in-cloud-computing