Load Balancing

By: **IBM Cloud Education**

10 June 2019

Networking

Load Balancing

In this guide, learn how load balancing optimizes website and application performance.

What is load balancing?

As strain increases on a website or business application, eventually, a single server cannot support the full workload. To meet demand, organizations spread the workload over multiple servers. Called "load balancing," this practice prevents a

single server from becoming overworked, which could cause it to slow down, drop requests, and even crash.

Load balancing lets you evenly distribute network traffic to prevent failure caused by overloading a particular resource. This strategy improves the performance and availability of applications, websites, databases, and other computing resources. It also helps process user requests quickly and accurately.

From a user perspective, load balancing acts as an invisible facilitator that sits between a client and a group of servers, ensuring connection requests don't get lost. Without load balancing, applications, websites, databases, and online services would likely fail when demand gets too high. A single high-traffic website may field hundreds or thousands of user requests at the same time. It needs multiple servers to accurately populate webpages with the requested information, including text, photos, video, and audio streaming.

You will often find load balancing in use at server farms that run high-traffic websites; it is also used for Domain Name System (DNS) servers, databases, and File Transfer Protocol (FTP) sites. If a single server handles too much traffic, it could underperform or ultimately crash. By routing user requests evenly across a group of servers, load balancers minimize the likelihood of downtime. They do this by rerouting traffic to other servers in the group if one should fail. When adding a new server to the server pool, a load balancer automatically includes it in the process of traffic distribution.

Load balancing acts as a "traffic cop," bringing order to a potentially chaotic situation. In certain environments, such as applications and virtual infrastructures, load balancing also performs health checks to ensure availability and prevent issues that can cause downtime. Load balancing can even provide centralized security across the group of servers that is easier to manage.

Load balancing performs these critical tasks:

- Manages traffic spikes and prevents spikes on a single server
- Minimizes user request response time
- Ensures performance and reliability of computing resources, both physical and virtual
- Adds redundancy and resilience to computing environments

How it works

The load balancer uses a predetermined pattern, known as a load balancing algorithm or method. This ensures no one server has to handle more traffic than it can process. Different algorithms manage the process using different techniques. You, therefore, have multiple options to choose from when making a decision on what type of load balancer to use.

Here are the basics of how a load balancer works:

1. A client, such as an application or browser, receives a request and tries to connect with a server.
2. A load balancer receives the request,and, based on the preset patterns of the algorithm, it routes the request to one of the servers in a server group (or farm).
3. The server receives the connection request and responds to the client via the load balancer.
4. The load balancer receives the response and matches the IP of the client with that of the selected server. It then forwards the packet with the response.
5. Where applicable, the load balancer handles SSL offload, which is the process of decrypting data using the Security Socket Layer encryption protocol, so that servers don't have to do it.
6. The process repeats until the session is over.

Benefits

If your organization runs high-traffic websites and applications or databases that receive a lot of queries, load balancing delivers multiple benefits by optimizing resource use, data delivery, and response time. In high-traffic environments, load balancing is what makes user requests go smoothly and accurately. They spare users the frustration of wrangling with unresponsive applications and resources.

Load balancing also plays a key role in preventing downtime and simplifying security, reducing the likelihood of lost productivity and lost profits for your organization.

Other benefits of load balancing include the following:

- **Flexibility:** Besides directing traffic to maximize efficiency, load balancing delivers the flexibility to add and remove servers as demand dictates. It also makes it possible to perform server maintenance without causing disruption for users since traffic gets rerouted to other servers during maintenance.
- **Scalability:** As the use of an application or website increases, the boost in traffic can hinder its performance if not managed properly. With load balancing,

you gain the ability to add a physical or virtual server to accommodate demand without causing a service disruption. As new servers come online, the load balancer recognizes them and seamlessly includes them in the process. This approach is preferable to moving a website from an overloaded server to a new one, which often requires some amount of downtime.

- **Redundancy:** In distributing traffic over a group of servers, load balancing provides built-in redundancy. If a server fails, you can automatically reroute the load to working servers to minimize the impact on users.

Application load balancing

Application load balancing performs the functions of classic load balancers by distributing user requests across multiple targets. But unlike traditional load balancers, which work only with IP addresses, application load balancers focus on content, taking into account URLs cookies and HTTP header content to determine which target to send each user request.

To implement application load balancing, developers code "listeners" into the application that to react to specific events, such as user requests. Listeners route the requests to different targets based on the content of each request (e.g, general requests to view the application, a request to load specific pieces of the application, etc.).

Application load balancers also perform health checks by periodically checking on each target to make sure it is not experiencing any issues. Based on the results, load balancers route traffic to healthy targets to ensure the user request is fulfilled instead of getting bogged down by an unhealthy target.

Hardware vs. software load balancing

When selecting a load balancer, you have a choice between a hardware and software version.

Hardware load balancers

Hardware load balancers consist of physical hardware, such as an appliance. These direct traffic to servers based on criteria like the number of existing connections to a server, processor utilization, and server performance.

Hardware load balancers include proprietary firmware that requires maintenance and updates as new versions and security patches are released. Hardware load balancers typically offer better performance and control—while offering a fuller range of features like Kerberos authentication and SSL hardware acceleration—but require some level of proficiency for proper management and maintenance. Because they are hardware-based, these load balancers are less flexible and scalable, so there is a tendency to over-provision hardware load balancers.

Software load balancers

Software load balancers usually are easier to deploy than hardware versions. They also tend to be more cost-effective and flexible, and they are used in conjunction with software development environments. The software approach gives you the flexibility of configuring the load balancer to your environment's specific needs. The boost in flexibility may come at the cost of having to do more work to set up the load balancer. Compared to hardware versions, which offer more of a closed-box approach, software balancers give you more freedom to make changes and upgrades.

Software load balancers can come in the form of prepackaged virtual machines (VMs). VMs will spare you some of the configuration work but may not offer all of the features available with hardware versions.

Software load balancers are available either as installable solutions that require configuration and management or as a cloud service—Load Balancer as a Service (LBaaS). Choosing the latter spares you from the routine maintenance, management, and upgrading of locally installed servers; the cloud provider handles these tasks.

Algorithms

To do their work, load balancers use algorithms—or mathematical formulas—to make decisions on which server receives each request. Algorithms vary and take into account whether traffic is being routed on the network or the application layer.

Load balancing algorithms fall into two main categories—weighted and non-weighted. Weighted algorithms use a calculation based on weight, or preference, to make the decision (e.g., servers with more weight receive more traffic). The algorithm takes into account not only the weight of each server but also the cumulative weight of all the servers in the group.

Non-weighted algorithms make no such distinctions, instead of assuming that all servers have the same capacity. This approach speeds up the load balancing process but it makes no accommodation for servers with different levels of capacity. As a result, non-weighted algorithms cannot optimize server capacity.

Methods

Each load balancing method relies on a set of criteria to determine which of the servers in a server farm gets the next request. There are five common load balancing methods:

- **Round Robin:** This is the default method, and it functions just as the name implies. The load balancer directs requests in a rotating fashion, with the first server in the group fielding a request and then moving to the bottom, where it awaits its turn to be called upon again. This technique ensures each server handles about the same number of connections.

- **Weighted Round Robin:** With this method, each server is assigned a weight (or preference), usually commensurate with its capacity. The higher the weight, the more requests a server receives. For instance, a server assigned a weight value of two receives double the requests of a server assigned a value of one.

- **Sticky Session:** Also known as session persistence, this method links specific clients and servers for the duration of a session. The load balancer identifies a user attribute either through a cookie or by tracking the user's IP address to make the link. Once the link is established, all requests from the user are sent to the same server until the session is over. This improves the user experience while also optimizing network resources.

- **Least Connections:** This method assumes all requests generate an equal amount of server load. As such, the server handling the lowest number of requests receives the next request that comes in.

- **IP Hash:** This algorithm creates a unique hash key based on both the source and destination IP address of the client and the server. The key is used to route the request and enables a dropped connection to be reestablished with the same server.

Load balancing in the cloud

Load balancing can either refer to the process of balancing cloud-based workloads or load balancers that are themselves based in the cloud. In a cloud environment, cloud balancing functions much the same as in other environments, except that it has to do with traffic related to a company's cloud-based workloads and their distribution across multiple resources, such as server groups and networks.

Balancing cloud workloads is just as important as balancing loads in any other context. The objective ultimately is high availability and performance. The better the workloads perform as a result of even traffic distribution, the less likely the environment is to suffer an outage.

Cloud-based load balancers are usually offered in a pay-as-you-go, as-a-service model that supports high levels of elasticity and flexibility. They offer a number of functions and benefits, such as health checks and control over who can access which resources. This depends on the vendor and the environment in which you use them. Cloud load balancers may use one or more algorithms—supporting methods such as round robin, weighted round robin, and least connections—to optimize traffic distribution and resource performance.

Docker Swarm load balancing

If you are running an environment with multiple Docker containers, you can benefit from applying load balancing to the containers. Docker is an open source development platform that uses containers to package applications for portability across systems running Linux. Docker Swarm is a clustering and scheduling tool employed by developers and administrators to create or manage a cluster of Docker nodes as a single virtual system.

In a Docker Swarm, load balancing balances and routes requests between nodes from any of the containers in a cluster. Docker Swarm has a native load balancer set up to run on every node to handle inbound requests as well as internal requests between nodes.

Depending on topology, you can set up a load balancer outside the Docker Swarm on its own single-node swarm to route requests that originate outside the cluster. The native Docker Swarm balancer is a basic Layer 4 version. As such, it may not provide all the features you require, such as SSL/TLS termination, access control, and authorization, content-based routing, as well as rewrites and redirects. Adding a balancer outside the cluster helps solve that problem.

For a deeper dive on Docker Swarm and how it compares to Kubernetes, see "Docker Swarm vs. Kubernetes: A Comparison."

Load balancing vs. clustering

Load balancing shares some common traits with clustering, but they are different processes. A cluster consists of a group of resources, such as servers, used for data storage or to run specific systems within your IT environment. Some clusters use the Kubernetes open source platform to schedule and distribute containers more efficiently.

Clustering provides redundancy and boosts capacity and availability. Servers in a cluster are aware of each other and work together toward a common purpose. But with load balancing, servers are not aware of each other. Instead, they react to the commands they receive from the balancer. You can employ load balancing in conjunction with clustering but load balancing also is applicable in cases involving independent servers that share a common purpose—to run a website, business application, web service, or some other IT resource.

Open source tools

An open source load balancer is a software load balancer you can download free of charge. Open source load balancers provide a desirable option if you are operating on a limited budget, but require you to have a tech-savvy development and operations team with the knowledge and skills to deploy and operate the solution and manage the necessary software licenses.

A variety of open source load balancers are available for download, each with different functionality and server compatibility. Open-source load balancers include Neutrino (link resides outside ibm.com), Gobetween (link resides outside ibm.com), LoadMaster by Kemp (link resides outside ibm.com), and the Linux-based Seesaw (link resides outside ibm.com). Some, such as LoadMaster and Neutrino, offer free versions and fee-based commercial versions with added functionality. If you're considering the open source route, be sure to review functionality and compatibility with your specific server when making a decision.

Load balancing and networks

When applied to networks, load balancing evenly distributes requests from servers and clients to other resources in the network. Network load balancers use the TCP/IP protocol to distribute traffic across wide area network (WAN) links. This supports network connection sessions such as email, web, and file transfers. By doing this, load balancing increases bandwidth to all users across the network. This ultimately improves responsiveness to their requests.

As with other load balancers, when a network load balancer receives a connection request, it chooses a target to make the connection. Some types of connections, such as when browsers connect to websites, require separate sessions for text, images, video, and other types of content on the webpage. Load balancing handles these concurrent sessions to avoid any performance and availability issues.

Network load balancing also provides network redundancy and failover. If a WAN link suffers an outage, redundancy makes it possible to still access network resources through a secondary link. The servers connected to a load balancer are not necessarily all in the same location, but if properly architected, this has no bearing on the load balancer's ability to do its work.

Network load balancing delivers business continuity and failover, ensuring that no matter how geographically dispersed your organization's workforce is, you will be able to maintain acceptable levels of availability and performance.

For an overview of networks, see "Networking: A Complete Guide."

IBM Cloud Internet Services provides more information about global load balancing.

VMware

In a VMware—or virtual—environment, you can set load balancing and failover policies to manage traffic distribution between network adapters. Adapters are the software procedures that establish connections between computers and networks, as well as between LANs and WANs. The load balancing policy directs a physical switch to control incoming traffic. A load balancing policy also controls outgoing

traffic. It does this by distributing the traffic among the network adapters linked to a switch or a port group on a virtual switch.

Databases

In a database context, load balancing concerns itself with the distribution of database queries, which are routed to multiple servers to maintain an even flow and avoid issues that can occur if there is a high number of concurrent requests. The benefits of database load balancing are identical to those in any other environment, such as an application, network, or Docker Swarm—including improved availability and performance and quicker response times.

Database load balancing contributes to data integrity by ensuring that queries do not fail before a transaction is completed. In the event of a failover, data integrity is maintained as the new primary server takes over the query without letting it die. This process increases the overall reliability of the database by minimizing errors that tend to occur as a result of server failure.

Load balancers and IBM Cloud

IBM offers several load balancers to help you improve uptime and maintain business continuity. IBM load balancers allow you to scale applications up or down by adding or removing servers without any major disruptions to service. This helps to ensure that users can access applications whenever they need them without problems:

- **IBM Cloud Load Balancer:** Delivered in an elastic as-a-service model, this load balancer provides a graphical interface to monitor server health and SSL offload for incoming traffic.
- **Citrix NetScaler® Appliances:** This load balancer platform delivers advanced features to support high-performance needs.
- **IBM Cloud Kubernetes Service LoadBalancer:** This load balancer was developed for use in conjunction with the IBM Cloud Kubernetes Service.

Explore the features and benefits of IBM's various load balancers.

Sign up for an IBMid and create your IBM Cloud account.

Why IBM Cloud

- Why IBM Cloud
- Hybrid Cloud approach
- Trust and security
- Open Cloud
- Data centers
- Case studies

Products and Solutions

- Cloud Paks
- Cloud pricing
- View all products
- View all solutions

Site feedback

Learn about

- What is Hybrid Cloud?
- What is Cloud Computing?
- What is Confidential Computing?
- What is a Data Lake?
- What is a Data Warehouse?
- What is Artificial Intelligence (AI)?
- What is Machine Learning?
- What is DevOps?
- What is Microservices?

Resources

- Get started
- Docs
- Architectures
- IBM Garage
- Training and Certifications
- Partners
- Cloud blog
- Hybrid Cloud careers
- My Cloud account

Let's talk