WHITE -PAPER

All About Load Balancing

Ensure continuity of service and improve performance of your Web applications

HAPROXY
Powering Your Uptime

American headquarters

375 Totten Pond Road, Suite 302

Waltham, Massachusetts 02451

USA

Phone: +1 (844) 222-4340

Email: contact@haproxy.com

European headquarters

15, Avenue Raymond Aron

92160 Antony

France

Phone: +33 1 30 67 60 74

Fax: +33 1 75 43 40 70

Email: contact@haproxy.com

TABLE OF CONTENT

WHITE-PAPER

ALL ABOUT LOAD BALANCING

Does your company have a website? An Intranet or Extranet? Web applications or services? Then this white-paper is for you. Welcome to the world of Load Balancing!

Context and Challenges

At the end of the 20th century, companies were discovering the importance of having a corporate website to provide information about their business, their products and/or services. A little more than ten years have passed since the Internet began to spread, and today, nearly everything can be done online. Remote access to information is everywhere, especially with the rise of Software as a Service applications and Cloud Computing services.

As practical and ef cient as the Internet "service oriented" model is, it also has its risks, including traf c congestion, decreased performance, service outages and security attacks. These risks can some times severely impact teams' productivity, brand image, and even cause a drop in sales.

Purpose of this White-Paper

The path from the client to the server is sometimes long and often full of pitfalls. The purpose of this white-paper is to give you the keys to understand load balancing for Web service's information ows.

This white-paper does not give a universal answer (each situation is unique), but will help application managers, IT managers and other corporate leaders tune their network and application architecture.

This white-paper covers load balancing tools and methods, as well as load balancing best practices based on your needs and constraints.

Icons Used in this White-Paper

Stop preconceived ideas! In these boxes, stop precon-ceived ideas. Once read, you cannot say that you had no idea!

Note!

The limitations, and sometimes even the drawbacks, of a given technique.

Remember!

What you need to remember...

To go deeper...

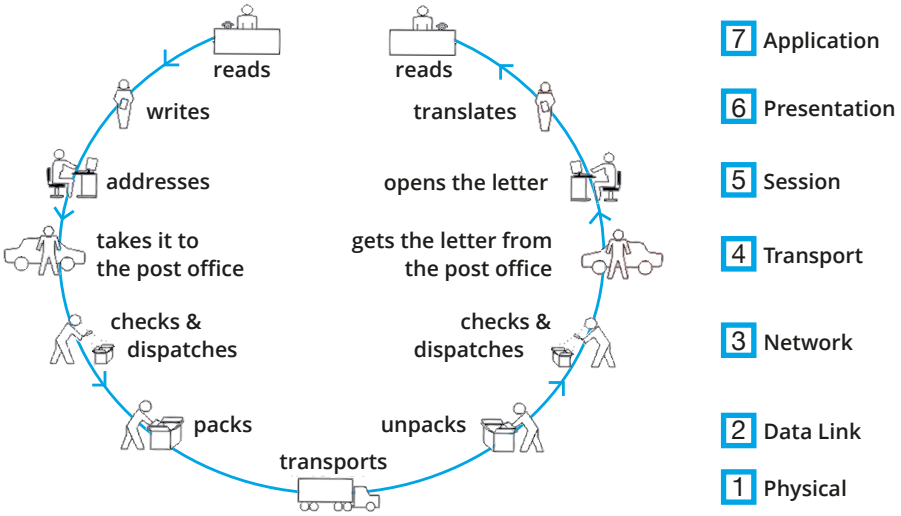
To increase your knowledge of the topic in question.

ARCHITECTURE, DEFINITIONS AND CONCEPTS

Typical Network Architecture

On one side a server. On the other a client station. Between the two a group of devices connected to one another so that information can flow from one to the other: The network.

A simple concept, but the network is more complicated than it appears. It is made up of many components, grouped into “layers” in the OSI model – the model on which a computer network’s operation is based.



Each component contributes, in its own way, to the smooth ow of information.
A load balancing solution can operate at two levels:

Layers 3/4: Network and transport (TCP/IP protocol)

Layer 7: Applications (HTTP, RDP, etc.)

In subsequent chapters, we will discuss in detail steps to take at each layer to improve performance and guarantee the availability of services that depend on this architecture.

Definitions and Concepts

Some basic information before getting down to business.

Packet: To make the ow of information easier, exchanged information is segmented into “packets.” Layer 4-Transport is responsible for packet segmentation and transmission, using Layer 3-Network.

Applications: The software programs used to exchange data, such as browsers, messaging clients, CRM tools and FTP transfer tools.

Load balancing: Load balancing is dividing the amount of work that a server has to do between two or more servers, so that more work gets done smoother and faster.

High availability: Guaranteeing users that applications have continuity of service, work properly and/or are accessible.

LOAD BALANCING... WITHOUT A LOAD BALANCER

The simplest load balancing method is to dedicate servers to predefined groups of users. While this method is easy to set up for an Intranet, it is extremely complex, even impossible, to set up for Internet servers.

In this case, specific parameterization of the DNS (Domain Name Server) at the network layer and/or application optimization (layer 7) are the first steps towards load balancing without a load balancer.

DNS (Domain Name Server)

In most situations, the round robin DNS method is a good start for load balancing. For the same specific server name, the DNS has multiple server IP addresses, that are sent to customers in rotating order. This method is completely transparent for users: they only see the website address. Round robin DNS is generally used for search engines, POP servers (messaging) or websites with static content.

Note!

Round robin DNS has two major drawbacks:

• It is not suitable for context management (if an application needs a user session opened);

• By itself, it does not have the means to check server availability; the DNS cache retains incorrect information if a server crashes or a website is inaccessible. Additional resources are needed to test the servers' status and switch from one to another if necessary.

Application Optimization

Application optimization often involves simple, but highly effective methods, especially when a load balancer is used.

Separation of static and dynamic content In most applications, dynamic content only accounts for approximately a quarter of content. For static objects, using a reverse proxy cache as a front end for the server farm enables servers to focus only on requests that involve dynamic content.

A task that can easily be performed by a light HTTP server (lighttpd or thttpd for example), and that does not necessarily require the installation of dedicated servers.

HTTP server optimization

Whatever Web server is used (Apache, IIS, etc.), its parameterization will play a key role in its level of load absorption.

To go deeper…

***Do not ignore this HTTP server optimization step!** It will greatly improve your applications' performance. See the manufacturer's specifications for the optimal configuration for your HTTP server.*

You can set up load balancing without a load balancer. The methods discussed above, however, do not provide any control over availability and therefore require additional resources to continually test the servers' statuses and switch traffic from a faulty server to another if necessary. These methods are not primary load balancing solutions, but have their merits and can be used in addition to setting up a load balancer.

THE LOAD BALANCER

Logically, setting up a load balancer is the most pertinent and effective way to set up load balancing for a population of users over several servers or even several sites.

The load balancer can be either a piece of hardware or a software program installed on a dedicated server or on application servers. This last option is the riskiest, particularly because of the risk of malfunction when deploying new components.

Based on requirements, a load balancer can operate at the network layer (layer 3/4) and/ or the application layer (layer 7).

Various Action Levels

Hardware or software, the load balancer is the additional layer that can optimize and regulate traffic, while relieving servers by balancing the load using predetermined algorithms (layers 3/4 and 7) and/ or using intelligent functions capable of taking into account the content of each request (layer 7).

Layer 3/4 Load Balancing (Network)

Layer 3/4 load balancing deals with the routing (TCP/IP) of network packets. The layer 3/4 load balancer gets involved when the TCP connection is opened, then routes the packets based on the selected algorithms, using 3 methods:

Direct routing

With this method, the balancer “deals the cards”: it distributes requests for the same address between local servers. The servers then respond to the client directly. This is Direct Server Return (DSR). Easy to set up because it does require any changes at the IP level, this method requires strong knowledge of TCP/IP for correct and optimal configuration. It also requires server proximity – the servers must be on the same network segment as the load balancer.

Tunneling

Tunneling is a change to direct routing that overcomes the server proximity problem by establishing tunnels between remote servers and the load balancer.

Network address translation (NAT)

In NAT, the load balancer centralizes all flows: requests (like in direct routing, the load balancer divides them among the servers), and responses. It “masks” the entire server farm, which does not need any special parameterization.

On the other hand, this method requires a very strict application configuration, in particular so that the servers do not send their internal addresses in responses. Additionally, this method significantly increases the load on the load balancer; it must convert addresses in both directions, maintain a session table and support the return traffic load.

Layer 7 Load Balancing (Application)

Taking on the application layer, the load balancer is no longer satisfied with "blindly" routing requests. It analyzes the content of each HTTP, RDP or other request to determine its routing.

Reverse proxy

In its traffic transmission role, the layer 7 load balancer acts as a reverse proxy and pretends to be the server. It is responsible for accepting connections to the client and establishing connections with the servers to transmit data (requests and responses).

With this method, users cannot directly connect to servers and a specific server configuration is not required.

The layer 7 load balancer requires more power than hardware solutions at the network layer. It, however, provides an initial level of security by only sending the server what it understands.

Transparent reverse proxy

Occasionally, layer 7 load balancing may come up against server integration (log monitoring or layer 3/4 firewalls) or protocol (FTP) constraints. In both cases, the load balancer can simulate the client's IP address as source for connections that it establishes with servers. This is a transparent reverse proxy.

A transparent reverse proxy requires a shut-off mode and servers must be configured not only to connect to clients via the load balancer, but also be located on a different network segment from clients.

Remember!

There are no good or bad choices for load balancing.

Deciding between a network layer load balancer and an application layer load balancer is a matter of distinct or cumulative needs. Two load balancers (layers 3/4 and 7) can coexist in the same infrastructure.

Stop preconceived ideas!

Thousands or millions of concurrent sessions?

Why, when talking about layer 3/4 load balancers, do we talk about supporting several million connections when layer 7 load balancers can only manage several thousand? Because at the TCP level (Network), the load balancer only sends requests that it receives to the server farm, by counting active and inactive sessions. It can therefore devote all its resources to this task. While an application load balancer (which manages HTTP connections) must view each request and analyze it, in order to send it to the proper server, a server that only counts active sessions. This task, which is more involved and eats up more resources, limits processing capacity for the number of concurrent connections.

Main Functions of a Load Balancer

As we have seen, a load balancer can be hardware or software. It can also operate at multiple layers: network or application. Nevertheless, a load balancer, regardless of type, will perform at least two main functions: balancing the load between servers and monitoring them.

Dynamically Balancing the Servers' Load

In practice, the load balancer uses balancing algorithms. There are multiple algorithms that meet a wide range of criteria: session length, changes in load, server capacities, etc. These algorithms include:

Round Robin

In Round Robin, the load balancer uses each server one after the other, in rotating order. Weighting can be added if the servers have different capacities (weighted round robin). This is the most effective method for Web servers.

Least Conn

This algorithm sends the new request to the server with the lightest load. Ideal for long sessions, this method is ill-suited for servers whose load varies from one second to another.

URL or IP address hashing

More deterministic, this method creates a more or less effective (under certain conditions) affinity between a client and a server.

Stop preconceived ideas!

Priority for the fastest server: False!

Sending the request to the server that responds fastest is not a pertinent solution because this can quickly create an imbalance within the server farm.

Stop preconceived ideas!

Priority for the server with the lightest load: False!

As previously seen, the Least Conn (Least Connections) algorithm systematically sends requests to the server with the lightest load. This method is completely ill-suited for Web servers, for example, where the load can vary from one second to another.

Monitoring Servers' Status

This is one of the most sensitive aspects of load balancing: if one of the servers crashes, all requests should be automatically balanced between the other servers in the farm, allowing users to experience continuous and uninterrupted access to the Web applications or services.

Called health checks, there are many methods for server monitoring such as a ping, a TCP connection attempt or sending an HTTP request (layer 7 only). A faulty server may sometimes respond to the ping, but not to TCP connections, or may accept these, but refuse to respond to HTTP requests. Sometimes even, for a multi-level Web application server, some HTTP requests will receive an immediate response, while others will fail.

This is why, to guarantee high availability for Web services, you must create a series of regular tests that are as relevant as possible, based on the type of application.

Note!

Server health checks require extremely granular settings, to find the best balance between server monitoring and performance dedicated to applications.

For server monitoring, software load balancers are by far the most flexible. They create automated scenarios that can be quickly changed and corrected in very little time.

To go deeper…

Encrypted flows (https)

Given the growing use of encrypted flows, many layer 7 load balancers support SSL protocol, acting as a reverse proxy and SSL terminal.

Note: Resource intensive, decrypting SSL requests can rapidly saturate the load balancer.

In this case, the best solution is to isolate SSL processing by having a farm of dedicated SSL reverse proxies. This makes things easier for both the load balancer and the Web servers that decrypt the requests.

"Intelligent" Functions of the Application Load Balancer

To meet certain specific needs, the layer 7 load balancer has additional functions due to its ability to analyze requests.

Persistence

Persistence is for situations where all requests from a single user must be sent to the same server for a given session, in order to save information related to said session (the user's "context"), such as shopping cart contents on an online retail website. These are persistent, or "sticky" sessions and load balancing with server affinity. Various methods can be used to set up persistence:

Redirection: This is one of the most economical solutions. It involves ensuring that the application redirects the request to the local address of the server in question (if the server crashes, the user will still be redirected to this server).

Session affinity: The balancer associates a user with a server. The simplest method is to identify the server to which the user's IP address connected during the last connection.

Cookie learning: When a user request arrives, the load balancer checks to see if the application cookie is present in the request's header and compares the cookie's value to its session table to redirect the user to the correct server. If there is no hit, the request will be sent to any server, using the selected algorithm. When the server responds to the client request, the load balancer will collect the server's identification information (by learning) to record in its session table. When the client's second request arrives, the saved information will be used to direct the client to the correct server.

Cookie insertion: Basically, the principle is the same as above, except that in this case the cookie is inserted directly onto the user machine by the load balancer (which no longer has to maintain a session table), when the server responds. Starting with the second request, the cookie is read by the load balancer and requests are immediately sent to the same server for the entire session.

Note!

Limitations of these methods:

Session affinity on the source IP address is not reliable because many users browse the Internet with variable IP addresses.

Cookie learning requires using a session table. If the main load balancer crashes, its backup will not be able to maintain the client-server association. Only synchronization of the session table will solve this problem. It is very difficult to keep a session table up to date between two load balancers.

While cookie insertion eliminates the limitations association with cookie learning, it is dependent on the user accepting cookies. The use of mobile terminals, sometimes limited to a single cookie, makes this method unusable. A workaround solution is to modify this method by changing the existing cookie and by prefixing it with the server's identifier.

Improving quality of service

Connection limits: To prevent saturating applications and servers, the load balancer limits the number of simultaneous connections.

Queue and buffer management: Located between client stations and servers, the load balancer regulates and optimizes traffic, absorbing load spikes and relieving servers by giving priority to those that are not saturated.

Dynamic switching: Based on the content, folders, host name, IP or port, the load balancer send the request to the appropriate server.

Protection against attacks

Filtering HTTP headers and protocol validation: Its ability to analyze requests enables the load balancer to check the validity of the request and to create rules for all fields in the HTTP header.

Protection against DoS and DDoS: By stepping in between the perpetrators of denial-of-service (DoS) and distributed denial-of-service (DDoS) attacks, the load balancer not only saves and slows down involved requests, but also simulates a success, by sending back a server error message.

List of IP access controls: Responsive, a layer 7 load balancer can help trace IP addresses responsible for an attack, thus automatically creating and/or modifying rules in order to block the attack.

HOW TO SELECT A LOAD BALANCER?

As we have seen, there are many load balancing solutions. They all rely on different tools and meet different needs.

Ask the Right Questions

Before selecting the best solution(s), analyze your needs and expectations from a load balancing solution.

Performance: How many simultaneous connections does the application in question require? This is a crucial point to ensure that the load balancer is not where the network is congested (a difficult situation to fix).

Reliability: Hardware? Software? On Web servers? On a dedicated server? Whatever the choice, the solution's reliability should be a priority because all traffic is supported by the load balancer.

Functions: Simply a router? Origin of requests? Requests need to be analyzed? Multiple sites? While functions are decisive factors, they are not critical when selecting a solution.

Note!

Manufacturers often state that they can perform all types of functions by using scripts, which are more akin to custom development. ...

The Best Solution: A Combination

Independently, each load balancing solution addresses a specific need. For comprehensive load balancing, the best solution is a multi-layer combination.

A first layer of network load balancers (hardware or software based on expected performance) will provide:

- Layer 3/4 load balancing between SSL reverse proxies and layer 7 load balancers;

- Monitoring of layer 7 load balancers. The second layer of application load balancers will provide the appropriate level of "intelligence," in particular when it comes to saving user context.

A combination is also the best solution to guarantee the scalability of the whole system. When the reverse proxies are saturated, it is easy to add additional devices until the layer 3/4 load balancers are saturated, which accomodates multi-gigabit networks.

Finally, if necessary, you can go further by using the round robin DNS method to address multiple layer 3/4 load balancers, which preferably are distributed on multiple sites.

Remember!

Note that in most cases, Web architectures and platforms are not really busy in relation to the technical capacities of current solutions. In this case, combined management of layers 3/4 and 7 may be provided by the same product, reducing the solution's total cost accordingly.

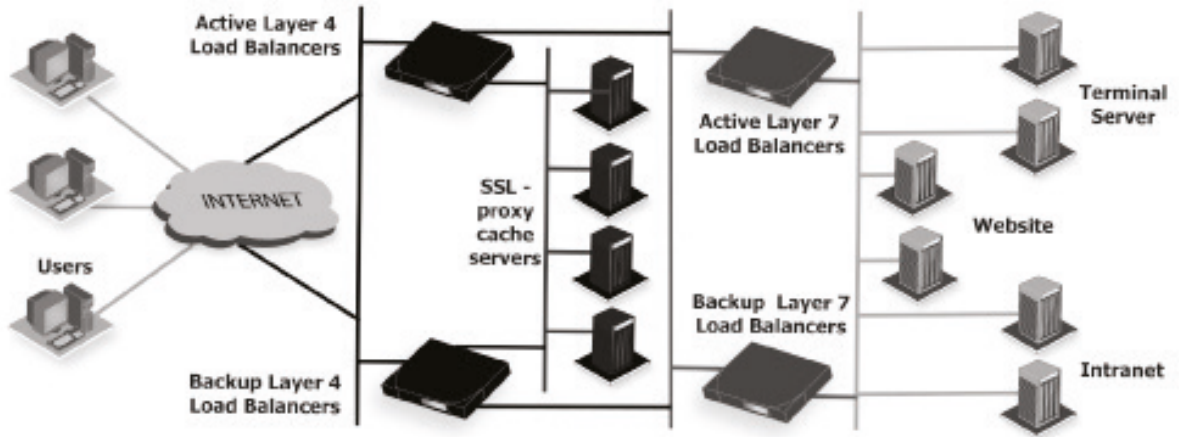
CONCLUSION

A decrease in response time or a service outage due to a saturated server (for load spikes) or temporarily unavailable server (for crash or maintenance)? Unable to stop production in the middle of the day? Set up a load balancing solution and these problems will become distant memories; the load balancer automatically sends requests to other servers to absorb the increase in traffic or so that repairs can be made.

Setting up a load balancer also means safer and more confident operation of the information system (knowledge of and control over the IS, anticipating needs, etc.) and optimization of existing resources.

Still, load balancing will not work miracles and cannot make up for the shortcomings and blocking points of a poorly written application (slow, costly in system resources, etc.) or of a poorly thought out architecture, which will invariably create problems for network flows.

To sum up, the ideal architecture could look like this:



"Segmented" as above, the architecture becomes more agile and scalable. It can easily be upgraded according to the company's additional needs.

Stop preconceived ideas…

The ideal architecture must be expensive: False!

Optimizing server performance using load balancing tools means that you can have lighter machines. They can still be administered even in greater numbers, are more flexible and require less energy than traditional servers.

ABOUT HAPROXY

Since its inception, HAProxy Technologies has offered a full line of load balancers to improve performance, guarantee quality of service and ensure the availability of critical business applications, by dynamically balancing flows and queries on the company's various servers.

Developed using HAProxy open source load balancing software, HAProxy Technologies solutions are known for their processing performance, reliability and wealth of features. Offered at more affordable prices than other commercial solutions, they cover 100% of the needs of 95% of companies and are easy to deploy and to administer.

HAProxy Technologies currently has hundreds of customers in the banking, retail industry, energy and e-commerce industries, and public sector. HAProxy Technologies products are also used by many hosting providers.

www.haproxy.com

EVERYTHING YOU EVER WANTED TO KNOW ABOUT LOAD BALANCING!

Does your company have a website? An Intranet or Extranet? Web applications or services? Then this white-paper is for you!

Today, nearly everything can be done online. A Web services tree that hides the infrastructure forest that is essential to absorb load spikes, guarantee optimal performance and prevent service outages.

METHODS, TOOLS AND BEST PRACTICES

The purpose of this white-paper is to give you the keys to understand load balancing and high availability for Web services and even to put an end to some preconceived notions.

You will (re)learn – through an overall picture – about the various methods, action levels and tools for load balancing, as well as their advantages, limitations and drawbacks based on the context.

This white-paper does not give a universal answer (each situation is unique), but it will help you choose application platform architecture.

LEARN ABOUT

Definitions and concepts

> Load balancing... without a balancer

> Action levels for a load balancer

> Main functions of a load balancer

> How to select a load balancer?

American headquarters

375 Totten Pond Road, Suite 302

Waltham, Massachusetts 02451

USA

Phone: +1 (844) 222-4340

Email: contact@haproxy.com

European headquarters

15, Avenue Raymond Aron

92160 Antony

France

Phone: +33 1 30 67 60 74

Fax: +33 1 75 43 40 70

Email: contact@haproxy.com