FullStack.Cafe - Kill Your Tech Interview

Q1: What Is Load Balancing? ☆

Topics: Software Architecture Load Balancing

Answer:

Load balancing is simple technique for distributing workloads across multiple machines or clusters. The most common and simple load balancing algorithm is Round Robin. In this type of load balancing the request is divided in circular order ensuring all machines get equal number of requests and no single machine is overloaded or underloaded.

The Purpose of load balancing is to

- Optimize resource usage (avoid overload and under-load of any machines)
- Achieve Maximum Throughput
- Minimize response time

Most common load balancing techniques in web based applications are

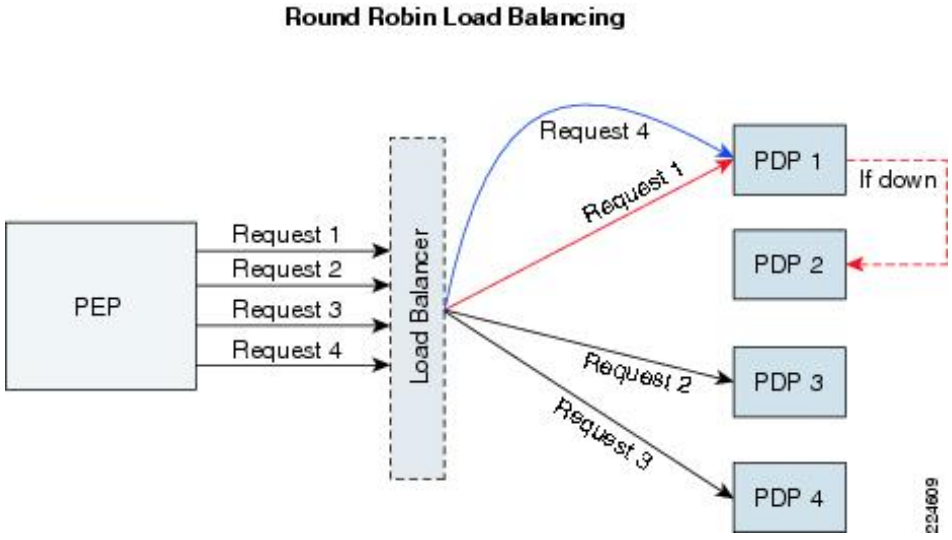
1. Round robin
2. Session affinity or sticky session
3. IP Address affinity

Q2: What Is Round-Robin Load Balancing? ☆☆

Topics: Load Balancing

Answer:

Round-robin load balancing is one of the simplest methods for distributing client requests across a group of servers. Going down the list of servers in the group, the round-robin load balancer forwards a client request to each server in turn. When it reaches the end of the list, the load balancer loops back and goes down the list again (sends the next request to the first listed server, the one after that to the second server, and so on).



Q3: Name some advantages of Round-Robin Load Balancing ☆☆

Topics: Load Balancing

Answer:

The main benefit of round-robin load balancing is that it is **extremely simple to implement.** However, it does not always result in the most accurate or efficient distribution of traffic, because many round-robin load balancers assume that all servers are the same: currently up, currently handling the same load, and with the same storage and computing capacity.

Q4: What Is Sticky Session Load Balancing? What Do You Mean By "Session Affinity"? ☆☆☆

Topics: Software Architecture Load Balancing

Answer:

Sticky session or a *session affinity technique* is another popular load balancing technique that requires a user session to be always served by an allocated machine.

In a load balanced server application where user information is stored in session it will be required to keep the session data available to all machines. This can be avoided by always serving a particular user session request from one machine. The machine is associated with a session as soon as the session is created. All the requests in a particular session are always redirected to the associated machine. This ensures the user data is only at one machine and load is also shared.

This is typically done by using SessionId cookie. The cookie is sent to the client for the first request and every subsequent request by client must be containing that same cookie to identify the session.

What Are The Issues With Sticky Session?

There are few issues that you may face with this approach

- The client browser may not support cookies, and your load balancer will not be able to identify if a request belongs to a session. This may cause strange behavior for the users who use no cookie based browsers.
- In case one of the machine fails or goes down, the user information (served by that machine) will be lost and there will be no way to recover user session.

Q5: What Is Load Balancing Fail Over? ☆☆☆

Topics: Software Architecture Load Balancing

Answer:

Fail over means switching to another machine when one of the machine fails. Fail over is a important technique in achieving high availability. Typically a load balancer is configured to fail over to another machine when the main machine fails.

To achieve least down time, most load balancer support a feature of *heart beat* check. This ensures that target machine is responding. As soon as a hear beat signal fails, load balancer stops sending request to that machine and redirects to other machines or cluster.

Q6: Why should we use Load Balancer (except preventing overloading)? ☆☆☆

Topics: Load Balancing

Answer:

Load balancers distribute incoming client requests to computing resources such as application servers and databases. In each case, the load balancer returns the response from the computing resource to the appropriate client. Load balancers are *effective* at:

- Preventing requests from going to *unhealthy servers*
- Preventing *overloading resources*
- Helping *eliminate single points of failure*

Additional benefits include:

- *SSL termination* - Decrypt incoming requests and encrypt server responses so backend servers do not have to perform these potentially expensive operations
- Removes the need to install *X.509 certificates on each server*
- *Session persistence* - Issue cookies and route a specific client's requests to same instance if the web apps do not keep track of sessions

Q7: What is the difference between Session Affinity and Sticky Session? ☆☆☆

Topics: Load Balancing

Answer:

Sticky session means that when a request comes into a site from a client all further requests go to the same server initial client request accessed. I believe that session affinity is a **synonym** for sticky session, but there are different ways of implementing it:

1. Send a cookie on the first response and then look for it on subsequent ones. The cookie says which real server to send to. **Bad** if you have to support cookie-less browsers
2. Partition based on the requester's IP address. **Bad** if it isn't static or if many come in through the same proxy.
3. If you authenticate users, partition based on user name (it has to be an HTTP supported authentication mode to do this).
4. Don't require state. Let clients hit any server (send state to the client and have them send it back) This is not a sticky session, it's a way to avoid having to do it.

I would suspect that sticky might refer to the cookie way, and that affinity might refer to #2 and #3 in some contexts, but that's not how I have seen it used (or use it myself).

Q8: What are some variants of Round-Robin Load Balancing algorithm? ☆☆☆

Topics: Load Balancing

Answer:

The following variants to the round-robin algorithm take additional factors into account and can result in better load balancing:

- **Simple Round-Robin** - Simplest methods for distributing client requests. It just forwards a client request to each server in turn. When it reaches the end of the list, the load balancer loops back and goes down the list again.
- **Weighted Round-Robin** – A weight is assigned to each server based on criteria chosen by the site administrator; the most commonly used criterion is the server's traffic-handling capacity. The higher the weight, the larger the proportion of client requests the server receives. If, for example, server A is assigned a weight of 3 and server B a weight of 1, the load balancer forwards 3 requests to server A for each 1 it sends to server B.
- **Dynamic Round-Robin** – A weight is assigned to each server dynamically, based on real-time data about the server's current load and idle capacity.

Q9: What is the Difference Between Weighted Load Balancing vs Round Robin Load Balancing? ☆☆☆

Topics: Load Balancing

Answer:

The biggest drawback of using the round robin algorithm in load balancing is that the algorithm assumes that servers are similar enough to handle equivalent loads. If certain servers have more CPU, RAM, or other specifications, the algorithm has no way to distribute more requests to these servers. As a result, servers with less capacity may overload and fail more quickly while capacity on other servers lie idle.

The weighted round robin load balancing algorithm allows site administrators to assign weights to each server based on criteria like traffic-handling capacity. Servers with higher weights receive a higher proportion of client requests. For a simplified example, assume that an enterprise has a cluster of three servers:

- Server A can handle 15 requests per second, on average
- Server B can handle 10 requests per second, on average
- Server C can handle 5 requests per second, on average

Next, assume that the load balancer receives 6 requests.

- 3 requests are sent to Server A
- 2 requests are sent to Server B
- 1 request is sent to Server C.

In this manner, the weighted round robin algorithm distributes the load according to each server's capacity.

Q10: What Is a TCP Load Balancer? ☆☆☆

Topics: Load Balancing

Answer:

A **TCP load balancer** is a type of load balancer that uses transmission control protocol (TCP), which operates at layer 4 — the transport layer — in the open systems interconnection (OSI) model. TCP load balancers are for applications **that do not use HTTP**. When deployed in front of a database cluster, a TCP load balancer spreads requests across all available server configurations.

Q11: Explain what is Reverse Proxy Server? ☆☆☆

Topics: Load Balancing Networking

Answer:

A Reverse Proxy Server, sometimes also called a reverse proxy web server, often a feature of a load balancing solution, stands between web servers and users, similar to a forward proxy. However, unlike the forward proxy which sits in front of users, guarding their privacy, the reverse proxy sits in front of web servers, and intercepts requests.

A reverse proxy server acts like a middleman, communicating with the users so the users never interact directly with the origin servers. It also balances client requests based on location and demand, and offers additional security. A reverse proxy cache server can:

- Enhance performance by caching local content
- Disguise the characteristics and existence of origin servers
- Carry TLS acceleration hardware, enabling them to perform TLS encryption in place of secure websites
- Distribute load from incoming requests to each of several servers (load balancing)
- Compress content, optimizing it and speeding loading times
- Perform multivariate testing and A/B testing without inserting JavaScript into pages
- Protection from DDoS attacks and related security issues

Q12: What Is IP Address Affinity Technique For Load Balancing? ☆☆☆☆

Topics: Software Architecture Load Balancing

Answer:

IP address affinity is another popular way to do load balancing. In this approach, the client IP address is associated with a server node. All requests from a client IP address are served by one server node.

This approach can be really easy to implement since IP address is always available in a HTTP request header and no additional settings need to be performed. This type of load balancing can be useful if you clients are likely to have disabled cookies.

However there is a down side of this approach. If many of your users are behind a NATed IP address then all of them will end up using the same server node. This may cause uneven load on your server nodes. NATed IP address is really common, in fact anytime you are browsing from a office network its likely that you and all your coworkers are using same NATed IP address.

Q13: Name some metrics for traffic routing ☆☆☆☆

Topics: Load Balancing

Answer:

Load balancers can route traffic based on various metrics, including:

- Random
- Least loaded
- Session/cookies
- Round robin or weighted round robin
- Layer 4
- Layer 7

Q14: What is the different between Layer7 vs Layer4 load balancing? ☆☆☆☆

Topics: Load Balancing

Answer:

- **Layer 4 load balancing** operates at the intermediate *transport* layer, which deals with delivery of messages with no regard to the content of the messages. Transmission Control Protocol (TCP) is the Layer 4 protocol for Hypertext Transfer Protocol (HTTP) traffic on the Internet. Layer 4 load balancers simply forward network packets to and from the upstream server without inspecting the content of the packets. They can make limited routing decisions by inspecting the first few packets in the TCP stream.

- **Layer 7 load balancing** operates at the high-level *application* layer, which deals with the actual content of each message. HTTP is the predominant Layer 7 protocol for website traffic on the Internet. Layer 7 load balancers route network traffic in a much more sophisticated way than Layer 4 load balancers, particularly applicable to TCP-based traffic such as HTTP. A Layer 7 load balancer terminates the network traffic and reads the message within. It can make a load-balancing decision based on the content of the message (the URL or cookie, for example). It then makes a new TCP connection to the selected upstream server (or reuses an existing one, by means of HTTP keepalives) and writes the request to the server.

Q15: What are the Pros and Cons of the "sticky session" load balancing strategy? ☆☆☆☆

Topics: Load Balancing

Answer:

If you *must use server-local session state* (e.g. shopping carts), sticky sessions are definitely the way to go. Your load balancer should be able to look at HTTP cookies (not only IP address) to determine stickiness, since IP addresses can change during a single session (e.g. docking a laptop between a wired and wireless network).

Pros:

- it's easy to implement - no app changes required.
- better utilizes local RAM caches (e.g. look up user profile once, cache it, and can re-use it on subsequent visits from same user)

Cons:

- if the server goes down, session is lost. (note that this is a con of storing session info locally on the web server-- not of sticky sessions per se). if what's in the session is really important to the user (e.g. a draft email) or to the site (e.g. a shopping cart) then losing one of your servers can be very painful.
- depending on "sticky" implementation in your load balancer, may direct unequal load to some servers vs. others
- bringing a new server online doesn't immediately give the new server lots of load-- if you have a dynamic load-balancing system to deal with spikes, stickiness may slow your ability to respond quickly to a spike. That said, this is somewhat of a corner case and really only applies to very large and sophisticated sites.
- if you have relatively few users but a single user's traffic can swamp one server (e.g. complex pages with SSL, AJAX, dynamically-generated images, dynamic compression, etc.), then stickines may hurt end-user response time since you're not spreading a single user's load evenly across servers. If you have a lot of concurrent users, this is a non-issue since all your servers will be swamped

Q16: What affect does SSL have on the way load balancing works? ☆☆☆☆

Topics: Load Balancing

Answer:

Using SSL in terms of load balancing, you get a couple of options:

- **Use a load-balancer that is your SSL/TLS endpoint**. In this case, the load-balancing will be done at the HTTP level: the client connects to the load-balancer and the load-balancer unwraps the SSL/TLS connection to pass on the HTTP content (then in clear) to its workers. In that case you don't need to install SSL cert on each app server/worker node.

- **Use a load-balancer at the TCP/IP level, which redirects entire the TCP connection directly to a worker node**. In this case, each worker node would have to have the certificate and private key (which isn't necessarily a problem if they're administered consistently). Using this technique, the load balancer doesn't do any HTTP processing at all (since it doesn't look within the SSL/TLS connection): on the one hand this reduces the processing done by the load-balancer itself, on the other hand it would prevent you from dispatching to a particular worker node based on the URL structure for example. Both methods have their advantages and disadvantages.

A disadvantages of the second method are:

1. the load balancer cannot see the HTTP request inside of all of that SSL and cannot add a header stating the real client IP; so to the web server, it will look as though all requests come from a single web client: the IP address of the load balancer itself.

2. the load balancer would not be able to use cookie based HTTP session affinity, since the cookies would be part of the encrypted data not visible to the load balancer. This means that different web servers would sometimes be getting HTTP requests for the same HTTP session, which has significant ramifications for the web server code

Q17: When to choose Round-Robin load-balancing method? ☆☆☆☆

Topics: Load Balancing

Answer:

The general consensus is that Round Robin works best **when the characteristics of the servers and requests are unlikely to cause some servers to become overloaded relative to others**. Some of the conditions are:

- All the servers have about the *same capacity.* This requirement is less important if differences between servers are accurately represented by server weights.
- All the servers host the *same content.*
- *Requests are pretty similar* in the amount of time or processing power they require. If there’s a wide variation in request weight, a server can become overloaded because the load balancer happens to send it a lot of heavyweight requests in quick succession.
- *Traffic volume is not heavy enough* to push servers to near full capacity very often. If servers are already heavily loaded, it’s more likely that Round Robin’s rote distribution of requests will push some servers “over the edge” into overload as described in the previous bullet.

Q18: What are some Load Balancing Algorithms you know? ☆☆☆☆

Topics: Load Balancing

Answer:

There is a variety of load balancing methods, which use different algorithms best suited for a particular situation.

- **Least Connection Method:** directs traffic to the server with the fewest active connections. Most useful when there are a large number of persistent connections in the traffic unevenly distributed between the servers.
- **Least Response Time Method:** directs traffic to the server with the fewest active connections, the fewest send requests and the lowest average response time.
- **Round Robin Method:** rotates servers by directing traffic to the first available server and then moves that server to the bottom of the queue. Most useful when servers are of equal specification, in a single geographic location and there are not many persistent connections.
- **IP Hash:** the IP address of the client determines which server receives the request.
- **The "Power of Two Choices"** - The algorithm decides which server will respond to each request by picking two *random* servers from the fleet and choosing the one with the fewest active connections. The whole purpose is to save the load balancer from *the cost of having to check all servers*, while still making a better choice than a purely random decision. By randomly picking a small number of entries among a list and then selecting the least loaded one, the probability of choosing an overloaded server decreases. This is especially true as the number of servers in the fleet grows and the distribution of selected servers widens. The system balances itself: The wider the distribution, the fairer the outcome.

Q19: What Are The Issues With Sticky Session? ☆☆☆☆

Topics: Load Balancing

Answer:

- The client browser may not support cookies, and your load balancer will not be able to identify if a request belongs to a session. This may cause strange behavior for the users who use no cookie based browsers.
- In case one of the machine fails or goes down (even during deployment), the user information (served by that machine) will be lost and there will be no way to recover user session.
- Sticky sessions don't allow our application to scale correctly. When we bind a server to a specific request, we directly remove the ability to use our other servers' capacity to fulfill consecutive requests.
- Binding sessions to specific servers can cause security concerns too. What if someone decides to create a session, and then perform a set of very CPU intensive requests on our application? By using sticky sessions, an attacker can perform this operation with half the resources that would be required if we were not using them.

Q20: What Is a UDP Load Balancer? ☆☆☆☆

Topics: Load Balancing

Answer:

A **UDP load balancer** is a type of load balancer that utilizes User Datagram Protocol (UDP), which operates at layer 4 — the transport layer — in the open systems interconnection (OSI) model. A UDP load balancing configuration is often used for live broadcasts and online games when speed is important and there is little need for error correction. UDP has low latency because it does not provide time-consuming health checks (that is a feature of TCP load balancer), which makes UDP the fastest option. Unlike the Transmission Control Protocol (TCP), UDP does not track data to ensure it was successfully transmitted and received.

FullStack.Cafe - Kill Your Tech Interview

Q1: Explain what is “Power of Two Random Choices” Load Balancing? ☆☆☆☆☆

Topics: Load Balancing

Answer:

- The **principle is** - The algorithm decides which server will respond to each request by picking two random servers from the fleet and choosing the one with the fewest active connections.
- The **purpose is** - Save the load balancer from the cost of having to check all servers, while still making a better choice than a purely random decision.

By randomly picking a small number of entries among a list and then selecting the least loaded one, the probability of choosing an overloaded server decreases. This is especially true as the number of servers in the fleet grows and the distribution of selected servers widens. The system balances itself: The wider the distribution, the fairer the outcome.

Q2: Compare UDP Load Balancer vs TCP Load Balancer ☆☆☆☆☆

Topics: Load Balancing

Answer:

TCP stands for Transmission Control Protocol. UDP stands for User Datagram Protocol. Load balancing methods are different for these protocols, and they represent the most common protocols used for sending data using Internet Protocol (IP).

The biggest difference between TCP and UDP load balancing is **how the data is tracked**:

- TCP guarantees the transmission and receipt of data by ordering and numbering the data packets. Then the recipient confirms delivery. A TCP load balancer also checks the data packets for errors. A TCP load balancer is considered the most *reliable* because data is tracked in transit to ensure no information is lost or corrupted.

- A UDP datagram only sends data. There is no confirmation of receipt. Nothing is tracked or checked for errors. All data, whether corrupted or not, simply arrives. If data is lost on UDP servers, there is no way to find it. Verifying all the data takes time and the lack of verification with a UDP load balancer allows for the fastest delivery of data possible. This speed is desirable for online games and live broadcasts.