Load Balancing

You cluster services in production environments to scale up applications and/or achieve high availability. By scaling up, the application can support more user requests and through high availability, the service is available even when a few servers are down.

You use a load balancer to distribute requests among the nodes in a cluster. The nodes that receive incoming traffic are a set of backend worker nodes. They are either pre-defined (static) or discovered dynamically. In the static mode, you cannot add new nodes to the pre-defined set of worker nodes at runtime. In the dynamic mode, you can add nodes to the load balancer at runtime without knowing the IPs and other connection details.

Among the many varieties of load balancers are hardware, DNS, transport-level (e.g., HTTP level like Apache Tomcat), and application-level load balancers (e.g., Synapse). High-level load balancers, like application-level load balancers, operate with more information about the messages they route and therefore, provide more flexibility but also incur more overhead. The choice of a load balancer is a trade-off between performance and flexibility.

There are many algorithms or methods for distributing the load between servers. Random or round-robin are simple approaches. More sophisticated algorithms consider runtime properties in the system like the machine’s load or the number of pending requests. The distribution can also be controlled by application-specific requirements like sticky sessions. With a reasonably diverse set of users, simple approaches perform as well as complex ones.

Session affinity

Stateful applications inherently do not scale well. State replication induces a performance overhead on the system. Instead of deploying stateful applications in a cluster, you can use session-affinity-based load balancing.

Session affinity ensures that, when a client sends a session ID, the load balancer forwards all requests containing the session ID to the same backend worker node, irrespective of the specified load balancing algorithm. Before the session is created, the request is dispatched to the worker node that is next-in-line and a session is established with that worker node.