Vertical Scaling vs Horizontal Scaling

Chapter 1 of Distributed Data

Businesses that predict growth in their user bases or have cyclical usage patterns can benefit from implementing a solution with computational resource scaling to avoid performance bottlenecks that would result in application slowness. In a scalable system, resources increase and decrease to meet changes in demand, which affects overall costs.

In today's data-driven and global business landscape, whether you are developing applications or evaluating ready-to-use industry solutions like Macrometa, a deep grasp of the underlying technology options is essential. As businesses encounter escalating data volumes and expand their operations, the criticality of scaling resources becomes increasingly evident.

There are two types of scaling that companies can implement: vertical and horizontal. Vertical scaling is the process of increasing or decreasing the capacity of

an existing resource such as the CPU and memory capacity of a server. Horizontal scaling is the process of adding or removing an additional resource to or from a cluster of resources. Each type of scaling has several important advantages that are explained in this article.

In traditional on-premises data centers, server hardware was specified to meet peak load demand, which led to inefficiency when systems operated below peak load. The peak load level was often merely an estimate calculated based on prior load data, so systems were over-specified, creating even more waste. In this model, if the system's demand increased beyond the limit of the available resources, an outage would occur.

Modern, cloud-based infrastructure systems provide resource scalability, which is often referred to as elasticity. Scaling is the ability to increase compute resources to match load increases and then reduce resources after the load decreases to enhance efficiency. Architecture should scale in a linear manner, where additional resources result in a proportional increase to serve additional load. A decrease in user activity creates a similar decrease in compute resource usage.

There are several types of IT systems that can be scaled, including, but not limited to, virtual machines, databases, storage, and containers.

Executive Summary

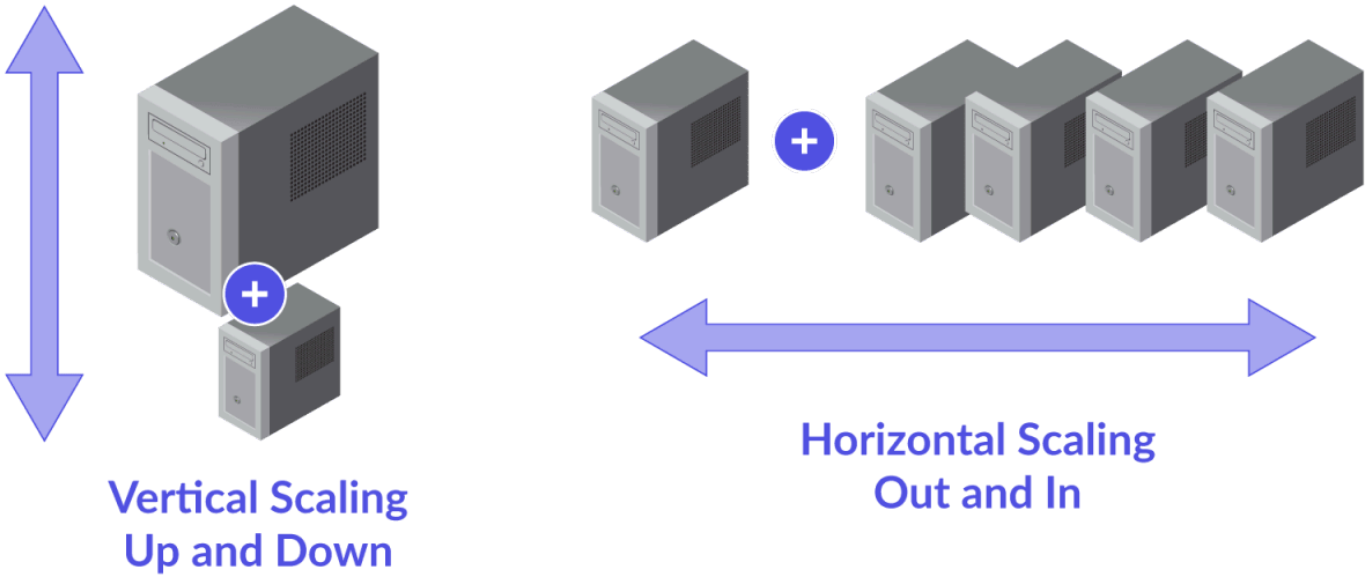
Enterprises need to understand the two types of scaling available in order to be equipped to deal with growth:

	Vertical Scaling	Horizontal Scaling
Description	Increasing or decreasing the capacity of an existing resource	Adding or removing an additional resource to or from a set or cluster of resources
Example	Adding/removing CPU or RAM to/from an existing virtual machine	Adding/removing virtual machines to or from a cluster of virtual machines

	Vertical Scaling	Horizontal Scaling
Scaling Operations (Increase/Decrease)	Scaling up / down	Scaling out / in
Efficiency	Suboptimal	Optimal
Required Architecture	Any	Distributed
Complexity	Low	High
Requires Downtime	Yes	No

Types of Scaling

There are two types of scaling: vertical and horizontal.



Vertical vs. Horizontal Scaling

Vertical Scaling

Vertical scaling is the process of increasing the capacity of an existing resource. Examples include adding CPU or RAM to an existing virtual machine or adding storage to an existing database instance.

Vertical scaling is simpler than horizontal because it does not require resources to be deployed in a distributed architecture the way horizontal scaling does.

The scaling dimension attributed to vertical scaling is scaling up and scaling down. Scaling up means increasing the capacity of a resource, while scaling down means reducing its capacity.

Horizontal Scaling

Horizontal scaling is the process of adding a resource to a set or cluster of resources. An example would be adding a virtual machine to a cluster of virtual machine clusters or adding a database to a database cluster.

Horizontal scaling requires resources to be deployed in a distributed architecture, commonly in a cluster, so that additional resources can be added to the system. Systems that are not designed to be operated in a cluster cannot be scaled horizontally.

The scaling dimensions attributed to horizontal scaling are scaling out and scaling in. Scaling out is the operation of adding additional resources to a cluster to increase capacity. Scaling in is the operation of removing resources from a cluster to decrease capacity.

Scaling Features

Resource Efficiency

With systems that do not scale, overprovisioned compute capacity is wasted when a system is not under peak load. Scaling enables the efficient operation of

is no need to overprovision compute resources to meet peak demand because the system capacity can scale to meet the demand.

Scaling Automation

Automating scaling operations is important because system administrators may not be available when system demand changes and scaling operations are required. There are several degrees of automation that can be applied to scaling.

Auto-Scaling

Auto-scaling is the ideal scaling scenario and an increase in load will trigger an automated increase in computing resources (scale up or out). The same scenario applies to scaling in or down, and a reduction in load will trigger an automated reduction in computing resources.

Manual Scaling

Manually scaling systems is the least desirable strategy, but it is an improvement over static resource provisioning. In this scenario, a system administrator manually performs scaling operations.

Scheduled Scaling

Various systems have predictable or cyclical load patterns (low/high load on the weekends, low/high load at night, etc.) and can be scheduled to scale based on the load cycle.

Distributed Databases

A distributed database is a clustered database system that enables the horizontal scaling of a database. A distributed database appears to a user as a single database, but it is a clustered set of multiple databases. The cluster management system ensures that the data is replicated among its databases and the data can be accessed simultaneously. Each database in the distributed database cluster cooperates to maintain the consistency of the data.

There are several technologies for managing a database cluster, and the choice is based on the type of database implemented in the system and ultimately is a function of the needs of the system and users. Popular examples include Oracle RAC, Apache Cassandra, MySQL NDB, Postgres PGCluster, Amazon DynamoDB, and hyper distributed cloud platforms such as Macrometa.

Comparison of Vertical and Horizontal Scaling

Vertical Scaling

On a virtual machine, vertical scaling is achieved by stopping an instance and resizing it to an instance type that has more RAM, CPU, I/O, or networking capabilities. Scaling vertically can eventually hit a limit, as a single virtual machine can only grow as large as the underlying hardware will allow. Vertical scaling operations can incur downtime, which decreases availability. However, vertical scaling is very easy to implement and can be sufficient for many use cases, especially in the short term.

Advantages of Vertical Scaling

- **Cost-Effective:** Adding resources to an existing server is trivially inexpensive, especially in a virtualized environment such as the cloud.
- **Less Complex System Functions:** When a single server (monolith) handles all services, it does not have to synchronize and communicate with other servers.
- **Less Complex Operations/Maintenance:** Maintenance costs are lower because there are fewer servers to manage.
- **Simpler Software:** Software can be developed as a monolith and does not need to be refactored into a distributed architecture.

Disadvantages of Vertical Scaling

- **Increased Downtime:** Single servers that are taken down for patching or upgrades create service outages.
- **Single Point of Failure:** Single servers increase the risk of losing data in the case of a hardware or software failure.
- **Hardware Limitations:** There is a limit to the resources that can be added to a single server. Every machine has its threshold for RAM, storage, and processing power.

Horizontal Scaling

Horizontal scaling is the optimal way to build applications to leverage cloud computing but requires applications to be developed in a manner that permits a distributed architecture. Systems that cannot be architected to distribute their workloads across multiple resources must be scaled vertically.

Advantages of Horizontal Scaling

- **Ease of Implementation:** Horizontal scaling is easier from a hardware perspective because to scale out you just add additional resources to your current pool.
- **High Availability / Redundancy:** A highly available system is fault-tolerant and can withstand the failure of an individual or multiple components (e.g., hard disks, servers, network links, etc.). The elimination of single points of failure increases resilience and creates high levels of service availability.
- **Canary Deployment:** A canary deployment is a deployment strategy in which a change is deployed to a subset of systems in a cluster rather than to all of its systems. This is an improvement to deploying changes to all systems in a cluster at once because a release engineer can monitor the subset of systems that had the change deployed. The engineer can then use this data and decide to roll back if there are errors or complete the full rollout if there are no errors.
- **Less Downtime:** There is no need to take the system down when load changes. You simply add or remove resources from the cluster pool as demand dictates.

- **Geographic Distribution:** Improved performance is achieved through reduced latency for global systems.
- **Improved Performance:** A distributed system can change capacity quickly without downtime to match demand, enabling the system to maintain a consistent level of performance and availability.

Disadvantages of Horizontal Scaling

- **Increased System Complexity:** A distributed/clustered architecture is more complex than a single server. Distributed systems must be engineered to synchronize and communicate with other servers.
- **Increased Complexity Operations/Maintenance:** A cluster of multiple servers is harder to maintain than a single server. This increases the number of systems that need to be managed and maintained. This architecture requires additional technologies to manage the cluster, such as load balancing, replication, and virtualization.

Scaling Strategy

Automated horizontal scaling is the ideal scaling strategy because it enables efficient scaling, improves performance, lowers downtime, improves resiliency, and optimizes cost. However, it is not suitable for all workloads. The system must be architected in a distributed cluster to take advantage of horizontal scaling.

The effort involved in setting up a system in a distributed architecture is often beyond the capacity and need of small, early-stage businesses. Vertical scaling is a simpler solution for businesses that do not have the staff to architect and operate a complex distributed system.

If market pressure is low or the system is already designed in a distributed architecture, automated horizontal scaling is the ideal scaling strategy.

Conclusion

Technology companies need to understand the types of scaling available, so they are equipped to deal with growth. Predicting the future compute resource needs of a business is a difficult task, so designing new systems to scale with demand is advantageous. Automated horizontal scaling is the ideal scaling strategy but is not suitable for all workloads: The system must be architected in a distributed cluster to take advantage of horizontal scaling.

Because cost is a function of scalable resources, efficient scaling architecture can drive benefits directly to the bottom line for technology companies. An alternative to managing the scaling of computing resources is to choose a hyper distributed cloud like Macrometa that offers serverless computing. A hyper distributed cloud is ideal for real-time uses cases with high performance results and actionable insights. Macrometa breaks the paradigm of centralized cloud computing and brings the computing resources to the edge of the network achieving a latency of less than 50 milliseconds when processing end-user queries.

This content was produced by Inbound Square

Chapters