Serverless computing

Serverless computing allows developers to purchase backend services on a flexible ‘pay-as-you-go’ basis, meaning that developers only have to pay for the services they use. This is like switching from a cell phone data plan with a monthly fixed limit, to one that only charges for each byte of data that actually gets used.

Serverless computing offerings typically fall into two groups, Backend-as-a-Service (BaaS) and [Function-as-a-Service (FaaS)](https://www.redhat.com/en/topics/cloud-native-apps/what-is-faas).

BaaS gives developers access to a variety of third-party services and apps. For instance, a cloud-provider may offer authentication services, extra encryption, cloud-accessible databases, and high-fidelity usage data. With BaaS, serverless functions are usually called through [application programming interfaces (APIs)](https://www.redhat.com/en/topics/api/what-are-application-programming-interfaces).

More commonly, when developers refer to serverless, they’re talking about a FaaS model. Under FaaS, developers still write custom server-side logic, but it’s run in containers fully managed by a cloud services provider.

The major public cloud providers all have one or more FaaS offerings. They include [Amazon Web Services](https://www.redhat.com/en/partners/aws) with AWS Lambda, [Microsoft Azure](https://www.redhat.com/en/partners/microsoft) with Azure Functions, [Google Cloud](https://www.redhat.com/en/partners/google) with multiple offerings, and [IBM Cloud](https://www.redhat.com/en/partners/ibm-alliance) with IBM Cloud Functions, among others.

What are the advantages of serverless computing?

* Lower costs - Serverless computing is generally very cost-effective, as traditional cloud providers of backend services (server allocation) often result in the user paying for unused space or idle CPU time.
* Simplified scalability - Developers using serverless architecture don’t have to worry about policies to scale up their code. The serverless vendor handles all of the scaling on demand.
* Simplified backend code - With FaaS, developers can create simple functions that independently perform a single purpose, like making an API call.
* Quicker turnaround - Serverless architecture can significantly cut time to market. Instead of needing a complicated deploy process to roll out bug fixes and new features, developers can add and modify code on a piecemeal basis.

What is Function-as-a-Service (FaaS)?

Function-as-a-Service (FaaS) is an event-driven computing execution model where developers write logic that is deployed in containers fully managed by a platform, then executed on demand.

In contrast to BaaS, FaaS affords a greater degree of control to the developers, who create custom apps rather than relying on a library of prewritten services.

Code is deployed into containers that are managed by a cloud provider. Specifically, these containers are:

* [Stateless](https://www.redhat.com/en/topics/cloud-native-apps/stateful-vs-stateless), making [data integration](https://www.redhat.com/en/topics/integration) simpler.
* Ephemeral, allowing them to be run for a very short time.
* Event-triggered, so they can run automatically when needed.
* Fully managed by a cloud provider, so that you only pay for what is needed, not always-on apps and servers.

Using FaaS, developers can call serverless apps through APIs which the FaaS provider handles through an [API gateway](https://www.redhat.com/en/topics/api/what-does-an-api-gateway-do).

 Each function will perform a specific task when triggered by an event, such as an incoming email or an HTTP request. After the customary stages of testing, developers then deploy their functions, along with their triggers, to a cloud provider account. When a function is invoked, the cloud provider either executes the function on a running server, or, if there is no server currently running, it spins up a new server to execute the function. This execution process is abstracted away from the view of developers, who focus on writing and deploying the application code.

Serverless architecture examples

Here are some common serverless architecture use cases:

* Trigger-based actions or running scheduled tasks (e.g., daily reports, backups, business logic, etc.)
* Building RESTful APIs for web and mobile applications
* Asynchronous processing (e.g., transcoding video)
* IT process automation, such as removing access automatically, initiating compliance security checks, or sending approvals
* Automating continuous integration and continuous delivery (CI/CD) pipelines (e.g., code commits triggering a build, pull requests triggering automated testing)
* Integrating with third-party services and APIs
* Running scheduled tasks (e.g., daily reports, backups, business logic, etc.)
* Real-time data processing for structured and unstructured data

\*\*\*Tools\*\*\*

AWS Amplify:

\* AWS Amplify is a set of tools and services provided by Amazon Web Services (AWS) for building full-stack serverless applications.

\* It offers libraries, CLI tools, and UI components for frontend development, as well as backend services like authentication, storage, and GraphQL APIs.

\* With AWS Amplify, developers can easily set up authentication, authorization, data storage, and real-time communication in their applications.

\* It provides seamless integration with other AWS services such as AWS Lambda, Amazon DynamoDB, Amazon S3, and Amazon API Gateway

Azure Functions:

\* Azure Functions is a serverless compute service provided by Microsoft Azure. It allows developers to build and deploy event-driven functions using various programming languages, including C#, JavaScript, Python, and Java.

\* Azure Functions supports triggers and bindings for integrating with various Azure services, such as Azure Blob Storage, Azure Cosmos DB, Azure Event Hubs, and Azure Service Bus.

\* It provides a rich development experience with features like local debugging, automatic scaling, and monitoring.

\* Azure Functions seamlessly integrates with other Azure services and tools such as Azure DevOps, Azure CLI, and Azure Portal.

Firebase Functions:

\* Firebase Functions is a serverless compute service provided by Firebase, a mobile and web application development platform acquired by Google.

\* It allows developers to write and deploy serverless functions that automatically scale in response to events from Firebase services such as Firebase Authentication, Firebase Realtime Database, and Firebase Cloud Messaging.

\* Firebase Functions supports multiple programming languages, including Node.js, Python, Go, and Java.

\* It provides a seamless development experience with features like local testing, automatic deployment, and integration with Firebase hosting and other Firebase services

Google Cloud Functions:

\* Google Cloud Functions is a serverless compute service offered by Google Cloud Platform. It allows developers to write and deploy event-driven functions using languages like Node.js, Python, Go, and Java.

\* Google Cloud Functions supports triggers for integrating with various Google Cloud services, including Cloud Storage, Cloud Pub/Sub, Cloud Firestore, and Firebase Realtime Database.

\* It offers features such as automatic scaling, built-in monitoring, and integration with Google Cloud Build for CI/CD workflows.

\* Google Cloud Functions integrates seamlessly with other Google Cloud Platform services and tools like Google Cloud Console, Google Cloud SDK, and Google Cloud APIs.

Quantum and edge computing

Edge computing uses localized processing of data to reduce the amount of data that must travel across the internet. This approach is a good fit for a wide range of applications, from real-time analytics to managing Internet of Things devices. Edge computing is also advantageous for data protection, as it can ensure the integrity of stored data. For example, it can aggregate data, split a single feed between multiple users and protect sensitive information.

The linear architecture of traditional computers has led to the development of alternate processing approaches. However, as data grows in size, response time to queries and requests is a slow process. The combined power of edge computing and quantum computing may help increase data processing speed. As quantum computing is more powerful than a classical computer processing, it can solve integer factorization faster than conventional computers. Edge computing can also reduce the amount of bandwidth used to transmit data across networks.

### Advantages

* Fast data processing, analysis, and reaction times provided by edge computing technologies enable real-time services. Rapid feedback is essential in automated driving, intelligent manufacturing, video monitoring, and other location awareness applications, which is why it offers consumers a choice of fast response services. For instance, real-time computer vision applications are made possible by edge computing.
* On-device computing lowers the quantity of data sent over the network, lowers the cost of transmission and the demand on the network’s capacity, lowers the energy used by local equipment, and increases computing effectiveness.
* Applications that benefit from a quicker response time, such as augmented reality and virtual reality, benefit from computing at the edge.
* The use of edge computing technologies can increase the stability, sturdiness, and accessibility of services. In mission-critical applications where network disconnections might have disastrous repercussions, strong dependability of linked on-device systems is crucial (e.g., medical monitoring or transportation systems).
* Edge computing can reduce network expenses, circumvent bandwidth restrictions, speed up data transmission, stop service outages, and offer you more control over the flow of critical data. Both dynamic and static caching are possible due to the decreased load times and greater proximity of online services to users.
* The services that use edge computing are more trustworthy, speedier, and less expensive. Customers benefit from a quicker, more reliable experience thanks to edge computing. Edge refers to low-latency, highly available apps with real-time service providers and company monitoring.
* Disadvantages
* A significant problem with edge computing is its cost. Without a local edge partner, building the infrastructure is expensive and difficult. The crew must maintain several gadgets in top condition at multiple locations, which results in frequent high maintenance costs.
* The entire attack surface of a network is increased via edge computing. Edge devices can be a point of entry for cyberattacks, giving an attacker the chance to introduce malicious software and infect the network.
* Unfortunately, creating strong security in a distributed environment is difficult. The majority of data processing takes place away from the security team’s and the central server’s direct line of sight. The attack surface grows as the company purchases new machinery.

Quantum computing is a multidisciplinary field comprising aspects of computer science, physics, and mathematics that utilizes quantum mechanics to solve complex problems faster than on classical computers. The field of quantum computing includes hardware research and application development. Quantum computers are able to solve certain types of problems faster than classical computers by taking advantage of quantum mechanical effects, such as superposition and quantum interference. Some applications where quantum computers can provide such a speed boost include machine learning (ML), optimization, and simulation of physical systems. Eventual use cases could be portfolio optimization in finance or the simulation of chemical systems, solving problems that are currently impossible for even the most powerful supercomputers on the market.

* Quantum bits, or qubits, are represented by quantum particles. The manipulation of qubits by control devices is at the core of a quantum computer's processing power. Qubits in quantum computers are analogous to bits in classical computers. At its core, a classical machine's processor does all its work by manipulating bits. Similarly, the quantum processor does all its work by processing qubits.
* In classical computing, a bit is an electronic signal that is either on or off. The value of the classical bit can thus be one (on) or zero (off). However, because the qubit is based on the laws of quantum mechanics it can be placed in a superposition of states.
* How do companies use quantum computing?
* Quantum computing can revolutionize industries. We give some example use cases below:
* **ML**
* Machine learning (ML) is the process of analyzing vast quantities of data to help computers make better predictions and decisions. Research in quantum computing studies the physical limits of information processing and is breaking new ground in fundamental physics. This research leads to advances in many fields of science and industry, such as chemistry, optimization, and molecular simulation. It is also a growing area of interest for financial services to predict market movements and for manufacturing to improve operations.
* **Optimization**
* Quantum computing can improve research and development, supply-chain optimization, and production. For example, you could apply quantum computing to decrease manufacturing process–related costs and shorten cycle times by optimizing elements such as path planning in complex processes. Another application is the quantum optimization of loan portfolios so that lenders can free up capital, lower interest rates, and improve their offerings.
* **Simulation**
* The computational effort required to simulate systems accurately scales exponentially with the complexity of drug molecules and materials. Even using approximation methods, current supercomputers cannot achieve the level of accuracy that these simulations demand. Quantum computation has the potential to solve some of the most challenging computational problems faced in chemistry, allowing the scientific community to do chemical simulations that are intractable today. For example,[Pasqal](https://aws.amazon.com/blogs/quantum-computing/quantum-chemistry-with-qucos-qubec-on-amazon-braket/) built their QUBEC computational software to run chemistry simulations. QUBEC automates the heavy lifting necessary to run quantum computational tasks from automatic provisioning of the computing infrastructure to running pre- and post-processing classical calculations and performing error mitigation tasks.

AI in the Cloud

Some of the more common AI-based applications in the cloud include:

* IoT – Cloud architectures and services that power the internet of things can store and process data generated by AI platforms on IoT devices.
* Chatbots – Chatbots are ubiquitous AI-based software that use natural language processing to carry out conversations with users – a boon for customer service in the age of instant gratification. Cloud platforms store and process the data captured by chatbots and cloud services connect them to the appropriate applications for further processing. Customer data is also fed back into the chatbot application that resides in the cloud.
* Business Intelligence –  BI is another mainstream application where AI cloud computing can gather data on the market, target audience, and competitors of customers. The cloud again facilitates the storage and transfer of data while the AI runs it through predictive analytics models.
* AI as a Service (AIaaS) – Public cloud vendors now offer AI outsourcing services, allowing companies to test out software and ML algorithms without risking their primary infrastructure. They can deploy off-the-shelf AI applications at a fraction of the cost of in-house AI with significant CAPEX savings.
* Cognitive cloud computing – Cognitive computing is the use of AI models to replicate and simulate human thought processes in complex situations. Players such as IBM and Google have built cognitive cloud platforms that provide cognitive insights-as-a-service to enterprises and facilitate the application of this technology in finance, retail, healthcare, and other industries.

Cloud networking

Content Delivery Network ( example Amazon CloudFront)

A content delivery network (CDN) is a network of interconnected servers that speeds up webpage loading for data-heavy applications. CDN can stand for content delivery network or content distribution network. When a user visits a website, data from that website's server has to travel across the internet to reach the user's computer. If the user is located far from that server, it will take a long time to load a large file, such as a video or website image. Instead, the website content is stored on CDN servers geographically closer to the users and reaches their computers much faster.

The primary purpose of a content delivery network (CDN) is to reduce latency, or reduce the delay in communication created by a network's design. Because of the global and complex nature of the internet, communication traffic between websites (servers) and their users (clients) has to move over large physical distances. The communication is also two-way, with requests going from the client to the server and responses coming back.  
  
A CDN improves efficiency by introducing intermediary servers between the client and the website server. These CDN servers manage some of the client-server communications. They decrease web traffic to the web server, reduce bandwidth consumption, and improve the user experience of your applications.

### Caching

Caching is the process of storing multiple copies of the same data for faster data access. In computing, the principle of caching applies to all types of memory and storage management. In CDN technology, the term refers to the process of storing static website content on multiple servers in the network. [Caching in CDN](https://aws.amazon.com/caching/cdn/) works as follows:

1. A geographically remote website visitor makes the first request for static web content from your site.
2. The request reaches your web application server or origin server. The origin server sends the response to the remote visitor. At the same time, it also sends a copy of the response to the CDN POP geographically closest to that visitor.
3. The CDN POP server stores the copy as a cached file.
4. The next time this visitor, or any other visitor in that location, makes the same request, the caching server, not the origin server, sends the response.

### Dynamic acceleration

Dynamic acceleration is the reduction in server response time for dynamic web content requests because of an intermediary CDN server between the web applications and the client. Caching doesn't work well with dynamic web content because the content can change with every user request. CDN servers have to reconnect with the origin server for every dynamic request, but they accelerate the process by optimizing the connection between themselves and the origin servers.

If the client sends a dynamic request directly to the web server over the internet, the request might get lost or delayed due to network latency. Time might also be spent opening and closing the connection for security verification. On the other hand, if the nearby CDN server forwards the request to the origin server, they would already have an ongoing, trusted connection established. For example, the following features could further optimize the connection between them:

* Intelligent routing algorithms
* Geographic proximity to the origin
* The ability to process the client request, which reduces its size

## What are the benefits of CDNs?

Content delivery networks (CDNs) provide many benefits that improve website performance and support core network infrastructure. For example, a CDN can do the following tasks:

### Reduce page load time

Website traffic can decrease if your page load times are too slow. A CDN can reduce bounce rates and increase the time users spend on your site.

### Reduce bandwidth costs

Bandwidth costs are a significant expense because every incoming website request consumes network bandwidth. Through caching and other optimizations, CDNs can reduce the amount of data an origin server must provide, reducing the costs of hosting for website owners.

### Increase content availability

Too many visitors at one time or network hardware failures can cause a website to crash. CDN services can handle more web traffic and reduce the load on web servers. Also, if one or more CDN servers go offline, other operational servers can replace them to ensure uninterrupted service.

### Improve website security

Distributed denial-of-service (DDoS) attacks attempt to take down applications by sending large amounts of fake traffic to the website. CDNs can handle such traffic spikes by distributing the load between several intermediary servers, reducing the impact on the origin server.

Load balancer:

Load balancing is the method of distributing network traffic equally across a pool of resources that support an application. Modern applications must process millions of users simultaneously and return the correct text, videos, images, and other data to each user in a fast and reliable manner. To handle such high volumes of traffic, most applications have many resource servers with duplicate data between them. A load balancer is a device that sits between the user and the server group and acts as an invisible facilitator, ensuring that all resource servers are used equally.

## What are the benefits of load balancing?

Load balancing directs and controls internet traffic between the application servers and their visitors or clients. As a result, it improves an application’s availability, scalability, security, and performance.

### Application availability

Server failure or maintenance can increase application downtime, making your application unavailable to visitors. Load balancers increase the fault tolerance of your systems by automatically detecting server problems and redirecting client traffic to available servers. You can use load balancing to make these tasks easier:

* Run application server maintenance or upgrades without application downtime
* Provide automatic disaster recovery to backup sites
* Perform health checks and prevent issues that can cause downtime

### Application scalability

You can use load balancers to direct network traffic intelligently among multiple servers. Your applications can handle thousands of client requests because load balancing does the following:

* Prevents traffic bottlenecks at any one server
* Predicts application traffic so that you can add or remove different servers, if needed
* Adds redundancy to your system so that you can scale with confidence

### Application security

Load balancers come with built-in security features to add another layer of security to your internet applications. They are a useful tool to deal with distributed denial of service attacks, in which attackers flood an application server with millions of concurrent requests that cause server failure. Load balancers can also do the following:

* Monitor traffic and block malicious content
* Automatically redirect attack traffic to multiple backend servers to minimize impact
* Route traffic through a group of network firewalls for additional security

### Application performance

Load balancers improve application performance by increasing response time and reducing network latency. They perform several critical tasks such as the following:

* Distribute the load evenly between servers to improve application performance
* Redirect client requests to a geographically closer server to reduce latency
* Ensure the reliability and performance of physical and virtual computing resources