Engineer

AWS

Amazon's cloud platform

What it is

AWS, short for Amazon Web Services, is Amazon's business of renting out computing by the hour and the gigabyte: virtual servers, file storage, managed databases, private networks, queues and a long catalogue of further services, all switched on through one API and billed to one account. Amazon has sold it since 2006. Two of its best-known services explain much of the rest. Amazon EC2 rents virtual machines, which AWS calls instances, and Amazon S3 stores files as objects inside containers called buckets.

In the vocabulary of the NIST definition, AWS is a public cloud: anyone with a payment card can open an account, resources appear within minutes, and every second or gigabyte of use is metered. It is usually weighed against Microsoft's Azure and Google Cloud, which sell much the same building blocks under other names. What AWS calls an EC2 instance is an Azure virtual machine or a Compute Engine instance on Google Cloud.

Every AWS resource lives somewhere specific. The service is divided into Regions, each designed to be isolated from the others, and every Region holds several Availability Zones. A zone is one or more discrete data centres with redundant power, networking and connectivity, physically separate from the other zones in its Region. A resource is created in one Region, often in a single zone, and a team chooses the place with latency, law and price in mind.

Access to all of it runs through AWS Identity and Access Management, or IAM. An account starts with a root user that can do anything, and IAM adds users, groups, roles and policies that say which API call is allowed on which resource under which conditions. Most questions about AWS security turn out to be questions about those policies.

Use the concept in a decision

Put “What is AWS?” to work in a real decision. We can help relate the concept to the system you are building or buying. Servers sized for what you actually use, moved into the cloud or back out of it, with the monthly bill worked out first.

Explore our services Get in touch →

Test the assumptions behind the term

Use the explanation and linked work to frame questions for your project. We can examine the assumptions with you before a tool is chosen. Ran 20 Docker apps on Google Cloud and AWS — 20 containerised applications deployed and watched

Ran 20 Docker apps on Google Cloud and AWS Get in touch →

Why it matters

Teams pick AWS for its range. A product that needs a managed PostgreSQL, a queue, a content delivery network, object storage and somewhere to run containers can have all of them under one bill and one identity system, documented to a common pattern. The range is also the price. Each service has its own pricing dimensions, its own limits and its own corner of the IAM policy language, and a team that adopts ten services now maintains ten small integrations with Amazon’s interfaces. Leaving later means replacing each one.

The second decision is what stays with the customer. AWS publishes a shared responsibility model and draws the line in two phrases: AWS is responsible for security “of” the cloud, meaning “the hardware, software, networking, and facilities that run AWS Cloud services”, and the customer for security “in” the cloud. Where the line falls depends on the service. On an EC2 instance the customer patches the guest operating system, looks after every program installed on it and configures the security group, the firewall in front of the instance. On S3 or DynamoDB, AWS runs the operating system and the platform, and the customer is left with the data, its encryption options and the permissions that decide who may read it.

The third is cost, which on AWS is a property of the design more than an invoice that arrives later. Amazon’s Well-Architected Framework names six pillars to review a workload against: operational excellence, security, reliability, performance efficiency, cost optimization and sustainability. Cost earns its place on that list because the same system can be built cheaply or expensively by choices made in its first week: instance sizes, how many zones, how much data leaves the Region, and which managed services stand in for a process on a plain server.

What it looks like in practice

On the tech stack page, AWS sits beside Google Cloud as one of the two clouds the company works in by default, with Azure added where a client already runs there.

One engagement used both of those clouds at once: twenty Docker applications ran across Google Cloud and AWS, some of them R-based analytics apps. They were deployed and maintained with the same configuration and monitoring in each cloud, so that no app turned into a special case of its own. When something broke, troubleshooting went through Podman and orchestration debugging through Kubernetes, and the analytics apps were kept stable and reproducible in production on both sides.

Another engagement ran the other way. For an international client, an AWS setup had been over-provisioned and had collected unused services, and the cost had grown out of proportion to the business's needs. The work reviewed which services were used, removed unnecessary services, adjusted the remaining capacity and moved workloads away from AWS. Those infrastructure changes are the delivered work described by the record. It does not establish a like-for-like cost ratio, a productivity measurement or a disruption-free migration. The example is useful for its method of examining the existing setup and changing what the business needs to run, not as a promise of a particular saving.

Both cases come down to the same habit. A pay-per-use service bills for whatever was switched on, so an AWS estate stays healthy when someone keeps checking what is running against what the business uses.

Where it goes wrong

AWS rarely loses data or stays down for long. The expensive incidents tend to start inside the customer's own account.

The root user does the day's work. Every account begins with a root user, the email address and password that opened it, with unrestricted access to every resource and setting, including closing the account. Signing in as root for routine tasks means that one leaked password reaches everything. AWS's guidance is to safeguard the root credentials, put multi-factor authentication on them and keep them for the handful of tasks that need them, while everyday work happens through federated sign-in and roles.

Long-lived access keys leak. An access key pair works until someone deletes it, and it travels well: into a laptop's dotfiles, a CI variable, a container image, a commit. The IAM best-practice list starts from the other end. People sign in through an identity provider and receive temporary credentials, and workloads take temporary credentials from an IAM role attached to the compute they run on. Where a long-term key is truly required, rotate it, and remove keys, users and roles that have gone unused.

A budget alert is taken for a brake. AWS Budgets sends a notification when actual or forecasted spend crosses a threshold, and its figures update up to three times a day, so an alert reports money that is already spent. The resources keep running and keep billing. Budgets can also take an action, such as applying an IAM policy or a service control policy that prevents new resources from being created, or acting on specific EC2 or RDS instances. It does so only when that action was configured in advance, with a role to act through.

Everything sits in one Availability Zone. A single instance with a single disk in one zone is the cheapest layout, and it shares the fate of that one location. Zones exist so that a power or network fault in one leaves the others running, and the AWS documentation recommends spreading an application across several. A database that matters gets a standby in a second zone, and a web tier gets instances in at least two, behind a load balancer that spans both.

A bucket opened for a demo stays open. New S3 buckets block public access by default, but a bucket policy or an object permission can change that in a minute, and a link shared with a client for one afternoon is still public a year later. S3 Block Public Access, set for the whole account or organisation, overrides those policies. Turn it on wherever an account has no reason to publish files, and serve public files from a bucket built for that purpose.

Further reading

- AWS Documentation — the index of user guides, API references and tutorials for every AWS service.
- Shared responsibility — the Well-Architected security pillar's account of which layers AWS secures and which the customer does.
- The pillars of the framework — the six areas AWS asks a workload to be reviewed against, cost and reliability among them.
- AWS Regions and Availability Zones — how Regions and zones are isolated, and why an application should span several zones.

- Security best practices in IAM — federation, roles, multi-factor authentication, the root user and least privilege, in one list.
- Managing your costs with AWS Budgets — budgets on actual and forecasted spend, their notifications and the actions they can run.

Glossary

Choose for your actual constraints

Bring the deployment path, the operational risks and the people on call. We can scope a change with checks, a recovery route and clear ownership.

See what it costs Get in touch →

Make the terminology practical

Ask how “What is AWS?” applies to your organisation. A concrete example from your current system is enough to start the discussion.

Explore our services

welcome@engineer.company +45 50 10 61 84 Copenhagen, Denmark

Take this page with you

Contact card

Carefully Crafted With Love by D.

CVR 45969703 · Tagensvej 105C, 3. th, 2200 København N

Copyright © 2025 – present · Engineer ApS · All rights reserved.

Printed from https://engineer.company/glossary/aws/