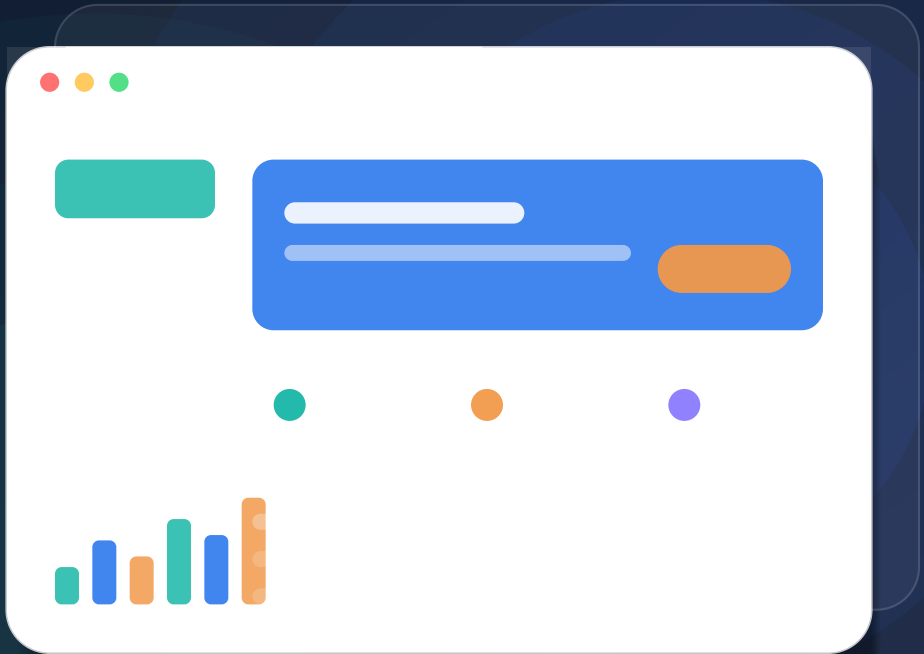


CLOUD COMPUTING FOR BEGINNERS

AWS • AZURE • GOOGLE CLOUD



**A Practical Introduction to
AWS, Azure and Google Cloud**

LEARN • BUILD • SECURE • SAVE

Cloud Computing for Beginners

A Practical Introduction to AWS, Azure and Google Cloud

This guide is provided for general information. Cloud products, prices, free offers and menus change often, so always check the provider's current documentation and pricing before relying on any detail. Cloud resources are billed by use, so set budget alerts and delete what you do not need.

Contents

- Introduction 4
- 1. What Is the Cloud? 6
- 2. Meet the Big Three 9
- 3. Creating Your Account Safely 12
- 4. Core Concepts Every Cloud Shares 16
- 5. Compute: Servers, Containers and Functions 20
- 6. Storage and Databases 24
- 7. Networking in the Cloud 28
- 8. Security and Identity 31
- 9. Understanding and Controlling Costs 35
- 10. Your First Projects 39
- 11. Choosing a Cloud and Growing Your Skills 43
- Appendix A: Checklists and Commands 46
- Appendix B: Glossary 48

Introduction

Almost everything you use online, from streaming and online banking to messaging and maps, runs in the cloud. Companies of every size now rent computing power, storage and software from a handful of giant providers instead of buying and running their own machines. Three of those providers dominate: **Amazon Web Services (AWS)**, **Microsoft Azure** and **Google Cloud Platform (GCP)**.

This book is a gentle, practical introduction to all three. It explains what the cloud is, how the major platforms are organised, and how to do the things beginners need to do: create a safe account, launch a server, store files, set up a database, understand networking and security, and keep costs under control. It does not assume any previous cloud or programming experience, although some comfort with using a computer and typing commands helps.

Rather than teaching one provider in depth, it shows how the same ideas appear on each, so that you learn the concepts once and can move between platforms with confidence.

How to use this book

- **Brand new to the cloud?** Read Chapters 1 to 4 in order.
- **Want to compare providers?** Chapter 2 has a side-by-side service table, and Chapter 11 helps you choose.
- **Ready to try something?** Chapter 10 walks through two first projects.
- **Worried about a surprise bill?** Read Chapter 9 before you create anything.
- **Need a quick lookup?** Appendix A has a command quick reference, and Appendix B has a glossary.

WARNING

Cloud services bill you for what you use, and mistakes can be expensive. Before experimenting, set a budget alert (Chapter 9), protect your account with multi-factor authentication (Chapter 8), and delete resources when you finish.

TIP

Cloud platforms change constantly. Product names, prices, free offers and menu layouts shift often. This book teaches concepts that last, and tells you where to check the latest details. Always confirm current terms on the provider's own pages.

What Is the Cloud?

"The cloud" simply means using computers owned and run by someone else, over the internet, and paying only for what you use. Instead of buying a server, housing it, powering it, cooling it and replacing it when it breaks, you rent the same capability by the second or the hour.

An everyday comparison

Think of electricity. You do not build a power station to run your house. You plug in and pay for the units you use. Cloud computing offers computing in the same way: switch it on when you need it, switch it off when you do not, and pay for the difference.

Why organisations move to the cloud

- **Speed.** A server that once took weeks to order and install is available in a minute or two.
- **Pay as you go.** No big upfront purchase. Costs follow usage.
- **Scalability.** Add capacity during busy periods and remove it afterwards.
- **Reach.** Run your application in data centres around the world, close to your users.
- **Reliability.** Providers build in backup power, networking and copies of data across locations.
- **Less maintenance.** The provider looks after the buildings, hardware and much of the underlying software.
- **Access to advanced tools,** such as machine learning, large-scale analytics and managed databases, without building them yourself.

The main service models

Cloud services come in layers. The higher you go, the less you manage yourself, and the less control you have.

Model	What you get	Example
IaaS (Infrastructure as a Service)	Raw building blocks: virtual servers, storage, networks. You manage the operating system and everything above it	A virtual machine
PaaS (Platform as a Service)	A ready platform to run your code. The provider manages servers and system software	A managed web app service
FaaS / serverless	You supply small pieces of code, the platform runs them on demand and bills per use	A function triggered by an upload
SaaS (Software as a Service)	Finished software used through a browser	Email, office apps, CRM

A common comparison: **IaaS** is renting a bare kitchen, **PaaS** is renting a kitchen with equipment where you just cook, and **SaaS** is eating at a restaurant.

Deployment models

- **Public cloud:** shared services from providers such as AWS, Azure and Google Cloud, with your resources isolated from other customers.
- **Private cloud:** cloud-style infrastructure used by a single organisation, often in its own data centre.
- **Hybrid cloud:** a mix of on-premises systems and public cloud working together.
- **Multi-cloud:** using more than one public cloud provider.

Trade-offs to know about

- **Costs can grow unexpectedly** if resources are left running or are poorly sized.
- **Security is shared.** The provider secures the platform, but you must secure your accounts, data and settings (Chapter 8).
- **Dependence on a provider.** Moving a complex system between clouds takes real effort.
- **Internet reliance.** You need a dependable connection.
- **A learning curve.** The number of services can be overwhelming at first.

TIP

You do not need to learn everything. Most real projects use a small set of building blocks: compute, storage, a database, a network and an identity system. Learn those well and the rest is detail.

Meet the Big Three

AWS, Azure and Google Cloud together provide the majority of the world's public cloud services. They offer broadly the same building blocks under different names. Learning the names is half of the challenge.

Amazon Web Services (AWS)

The oldest and, by most measures, the largest. AWS launched in 2006 and offers the widest catalogue of services. It has an enormous community, extensive documentation and the biggest job market. Its console and naming can feel dense to newcomers.

Microsoft Azure

Closely tied to Microsoft's other products, which makes it a natural choice for organisations that already use Windows Server, Microsoft 365, Active Directory (now called Microsoft Entra ID) and SQL Server. It is strong in hybrid setups and enterprise tooling.

Google Cloud Platform (GCP)

Built on the infrastructure that runs Google's own services. It is respected for data analytics, machine learning, networking and Kubernetes (which Google created). Its console and project model are often praised as clean and approachable for newcomers.

Service equivalents at a glance

This table maps common needs to each provider's service. Treat names as a starting point for searching, since products are renamed and re-launched from time to time.

Need	AWS	Azure	Google Cloud
Virtual machines	EC2	Virtual Machines	Compute Engine
Object storage	S3	Blob Storage	Cloud Storage
Disk for a VM	EBS	Managed Disks	Persistent Disk
Shared file storage	EFS	Azure Files	Filestore
Managed SQL database	RDS, Aurora	Azure SQL Database, Azure Database for PostgreSQL	Cloud SQL, AlloyDB
NoSQL database	DynamoDB	Cosmos DB	Firestore, Bigtable
Data warehouse	Redshift	Synapse, Fabric	BigQuery
Serverless functions	Lambda	Functions	Cloud Run functions
Serverless containers	Fargate, App Runner	Container Apps	Cloud Run
Managed Kubernetes	EKS	AKS	GKE
Private network	VPC	Virtual Network	VPC network
Load balancing	Elastic Load Balancing	Load Balancer, Application Gateway	Cloud Load Balancing
DNS	Route 53	Azure DNS	Cloud DNS
Content delivery	CloudFront	Front Door, CDN	Cloud CDN
Identity and access	IAM	Entra ID, Azure RBAC	Cloud IAM

Need	AWS	Azure	Google Cloud
Secrets and keys	Secrets Manager, KMS	Key Vault	Secret Manager, Cloud KMS
Messaging	SQS, SNS	Service Bus, Event Grid	Pub/Sub
Monitoring and logs	CloudWatch	Azure Monitor	Cloud Monitoring and Logging
Audit trail	CloudTrail	Activity Log	Cloud Audit Logs
Command-line tool	aws	az	gcloud

Which one is best?

There is no universal winner. The right choice depends on your skills, your existing technology, your budget and the specific services you need. Chapter 11 gives guidance. For learning purposes, any of the three is a fine start, because the ideas transfer.

TIP

Concepts transfer between providers far better than product names. If you understand what object storage, a virtual network or a role is, you can find the equivalent anywhere.

Creating Your Account Safely

Signing up is easy. Doing it safely is the part that beginners skip, and it is where most disasters begin. Spend an extra half hour here.

What you will need

- An **email address** you control and will keep. Consider a dedicated address for the account.
- A **phone number** for verification.
- A **credit or debit card**. All three providers ask for one when you sign up for a free account, mainly to confirm identity.
- An **authenticator app** on your phone for multi-factor authentication.

Free offers: what to expect

All three providers have introductory offers for new customers. The details change regularly, depend on your country and sign-up date, and have conditions. The summary below reflects the providers' own pages at the time of writing; always verify the current terms before you start.

Provider	Introductory offer	Ongoing free usage
AWS	A Free plan with \$100 in credits on sign-up, plus up to \$100 more by completing guided tasks. The plan runs up to 6 months or until credits run out, and credits expire after 12 months. If you do not upgrade to a paid plan, the account closes and its data is kept for a limited period (90 days at the time of writing) before deletion	Always-free allowances for many services, such as monthly Lambda requests and DynamoDB storage
Azure	\$200 credit to spend within 30 days, plus free monthly amounts of 20+ services for 12 months (for example 750 hours of certain small virtual machines). If you do nothing at the end, the account and services are disabled until you move to pay-as-you-go	Free monthly amounts of 65+ services, such as Functions and Container Apps
Google Cloud	\$300 free trial credit for new customers. You are not charged until you activate your full paid account	Always Free products with monthly limits, such as one e2-micro VM (in certain regions), 5 GB of Cloud Storage and 1 TB of BigQuery queries

WARNING

Free tiers have limits and conditions. Using services outside them, forgetting running resources, or choosing a large instance can produce a real bill. On the AWS Free plan, enabling certain organisation-level services can immediately end your credits and convert the account to paid, so read the warnings the console shows.

Secure your account in the first hour

1. **Create the account** with a strong, unique password, stored in a password manager.
2. **Turn on multi-factor authentication (MFA)** for the main (root or owner) account immediately, preferably with an authenticator app or a hardware security key.
3. **Do not use the root account for daily work.** On AWS, create an admin user or use IAM Identity Center. On Azure and Google Cloud, use your named user account with appropriate roles.
4. **Set a budget alert** so you are emailed if spending rises (Chapter 9).
5. **Record your recovery details** safely, such as backup codes.
6. **Choose a default region** close to you or your users.

Console, command line and code

You can work with every cloud in three ways:

- **The web console** is a browser dashboard. It is the easiest way to explore and learn.
- **The command-line interface (CLI)** lets you run commands, which is faster and repeatable.
- **Infrastructure as code and SDKs** let you describe resources in files or programs. This is how professionals work, because it is repeatable and reviewable (Chapter 4).

Installing the command-line tools

Provider	Tool	First command after install
AWS	AWS CLI	<code>aws configure</code>
Azure	Azure CLI	<code>az login</code>
Google Cloud	Google Cloud CLI	<code>gcloud init</code>

Each provider also offers a browser-based **cloud shell** with the tool already installed, so you can start without installing anything.

```
# Check who you are signed in as
aws sts get-caller-identity      # AWS
az account show                 # Azure
gcloud auth list                 # Google Cloud
```

TIP

Use the cloud shell for your first experiments. It avoids installation problems and keeps your credentials off your own computer.

Core Concepts Every Cloud Shares

Behind the different names, the three platforms are built from the same ideas. Understand these and every platform becomes familiar.

Regions and availability zones

- A **region** is a geographic area containing cloud data centres, such as "US East" or "West Europe".
- An **availability zone** (AWS and Google Cloud use "zone"; Azure uses "availability zones" within regions) is one or more separate data centres within a region, with independent power and networking.
- Place resources in a region **close to your users**, check that required services are available there, and consider data laws about where data may be stored.
- Spreading across several zones protects you if one data centre fails.
- Prices differ by region.

How accounts are organised

Each provider has a hierarchy for grouping resources, billing and permissions:

Level	AWS	Azure	Google Cloud
Top	Organization	Tenant and management groups	Organization
Billing and isolation	Account	Subscription	Project

Level	AWS	Azure	Google Cloud
Grouping	Tags, stacks	Resource group	Folders, labels
Individual item	Resource	Resource	Resource

In practice, a beginner works with **one account, subscription or project**. Using separate ones for experiments and for real work is a good habit, because it limits mistakes and cost.

Resources, tags and labels

Everything you create is a **resource**: a virtual machine, a bucket, a database. Give resources meaningful names and attach **tags** (AWS and Azure) or **labels** (Google Cloud) such as `project=blog` or `owner=maria`. They make it easy to find, group and charge for resources, and to clean up afterwards.

Pay as you go

Most services charge for some combination of **time** (per second or hour), **amount stored**, **data transferred out** and **requests made**. Resources that are switched off may still incur storage or address charges. Chapter 9 covers this in detail.

Scaling

- **Vertical scaling** means a bigger machine (more CPU or memory).
- **Horizontal scaling** means more machines sharing the load.
- **Autoscaling** adds and removes capacity automatically as demand changes.

High availability and backups

Hardware fails. Systems designed for the cloud assume this, and spread across zones, keep copies of data and replace failed parts automatically. Backups protect you from mistakes and attacks; replication protects you from hardware failure. They are not the same thing.

Infrastructure as code

Instead of clicking through a console, you describe your infrastructure in text files, and a tool creates it. The same files recreate it identically, and can be reviewed and stored in version control.

Tool	Notes
Terraform / OpenTofu	Works across all three clouds; popular and widely taught
CloudFormation, CDK	AWS-native
Bicep and ARM templates	Azure-native
Pulumi	Uses general-purpose programming languages across clouds

Everyday CLI patterns

```
aws s3 ls                # AWS: list buckets
az group list            # Azure: list groups
gcloud projects list     # GCP: list projects

gcloud config set project PROJECT_ID
az account set --subscription "NAME"
```

TIP

Make a habit of checking which account, subscription or project you are working in before you create or delete anything. Working in the wrong one is a classic beginner mistake.

Compute: Servers, Containers and Functions

Compute is the heart of the cloud: the processing power that runs your programs. There are three main styles, from most control to least.

1. Virtual machines (VMs)

A VM is a software-based computer you rent. You pick the size, operating system and location, and you get a full machine you can log in to, as if it were a physical server. You are responsible for updating it and securing it.

Choice	What it means
Image	The starting operating system and software, such as Ubuntu or Windows Server
Size or instance type	How many CPU cores and how much memory. Small is cheap
Disk	Storage attached to the VM, which can be fast (SSD) or cheaper
Network and firewall	Who may connect, and to which ports
Access	An SSH key (Linux) or password (Windows), plus a user account

Common starting sizes are small, burstable machines such as AWS t3.micro, Azure B1s and Google Cloud e2-micro. Check the free offers in Chapter 3 first.

Launching a VM from the command line

These examples show the idea on each cloud. Replace placeholders with your own values; image IDs and names change over time, and the console is a fine alternative.

```
# AWS: needs a real image ID and key pair name
aws ec2 run-instances \
  --image-id ami-0123456789abcdef0 \
  --instance-type t3.micro \
  --key-name my-key --count 1
```

```
# Azure
az group create --name demo-rg --location eastus
az vm create --resource-group demo-rg \
  --name demo-vm --image Ubuntu2204 \
  --size Standard_B1s \
  --admin-username azureuser \
  --generate-ssh-keys
```

```
# Google Cloud
gcloud compute instances create demo-vm \
  --zone=us-central1-a \
  --machine-type=e2-micro \
  --image-family=debian-12 \
  --image-project=debian-cloud
```

Connecting

```
ssh -i my-key.pem ec2-user@PUBLIC_IP    # AWS
ssh azureuser@PUBLIC_IP                  # Azure
gcloud compute ssh demo-vm               # Google
```

Stopping versus deleting

Stopping a VM usually stops the compute charge, but attached disks and reserved IP addresses may still cost money. **Deleting** the VM and its disk ends the cost. On Azure, a VM that is merely shut down from inside the operating system may still be billed; use the "stop (deallocate)" action in the portal or CLI.

```
aws ec2 terminate-instances --instance-ids i-0abc...
az group delete --name demo-rg
gcloud compute instances delete demo-vm \
  --zone=us-central1-a
```

2. Containers

A **container** packages an application with everything it needs, so it runs the same everywhere. Containers start fast and use fewer resources than VMs. Docker is the common tool for building them.

- **Serverless containers** (AWS App Runner or Fargate, Azure Container Apps, Google Cloud Run) run your container without you managing servers. They are an excellent beginner option.
- **Kubernetes** (AWS EKS, Azure AKS, Google GKE) orchestrates many containers across many machines. It is powerful but complex, and not where most beginners should start.

3. Serverless functions

With **functions** (AWS Lambda, Azure Functions, Google Cloud Run functions) you upload a small piece of code that runs in response to an event, such as a web request, a file upload or a timer. You pay only while it runs, and there are no servers to look after. They are well suited to short, event-driven tasks.

Choosing between them

If you want	Consider
Full control, or to run legacy software	Virtual machines
To run a web app or API with little effort	A serverless container service or an app platform
Small automation tasks and event handling	Serverless functions
Many containers at large scale	Managed Kubernetes

WARNING

A VM with a public IP address and open ports is quickly found by automated attackers. Restrict access (Chapter 7), keep the system updated, and never leave default or weak passwords in place.

Storage and Databases

Almost every application needs somewhere to keep data. Cloud providers offer several kinds of storage, each suited to a different job.

Three kinds of storage

Type	Best for	AWS / Azure / Google Cloud
Object storage	Files, images, videos, backups, website assets. Practically unlimited, accessed over the web	S3 / Blob Storage / Cloud Storage
Block storage	Disks attached to a virtual machine	EBS / Managed Disks / Persistent Disk
File storage	Shared network folders used by several servers	EFS / Azure Files / Filestore

Object storage in practice

Files are stored in a **bucket** (AWS and Google Cloud) or a **container** inside a **storage account** (Azure). Each file is an **object** with a name (key). Bucket names must be globally unique.

```
# AWS
aws s3 mb s3://my-unique-bucket-name
aws s3 cp photo.jpg s3://my-unique-bucket-name/

# Google Cloud
gcloud storage buckets create gs://my-unique-name \
  --location=us-central1
gcloud storage cp photo.jpg gs://my-unique-name/
```

```
# Azure
az storage account create --name myuniquestore123 \
  --resource-group demo-rg --location eastus \
  --sku Standard_LRS
az storage container create --name files \
  --account-name myuniquestore123 --auth-mode login
az storage blob upload --container-name files \
  --account-name myuniquestore123 --auth-mode login \
  --name photo.jpg --file photo.jpg
```

Storage classes and lifecycle

Data you read often belongs in a **standard** class. Rarely used data can go in cheaper **infrequent access** or **archive** classes, with lower storage prices but higher access costs and sometimes retrieval delays. **Lifecycle rules** move or delete data automatically as it ages, which is a simple way to cut cost.

Keeping storage safe

- **Keep buckets private by default.** Public buckets are a leading cause of data leaks.
- Turn on **versioning** to recover from accidental overwrites and deletions.
- Use **encryption**, which is on by default on all three platforms.
- Replicate important data to another region for resilience.
- **Back up** separately; replication alone copies mistakes as well.

Databases

You can install a database on a VM yourself, but **managed databases** are usually better: the provider handles installation, patching, backups, failover and scaling, so you focus on your data.

Kind	Good for	Examples
Relational (SQL)	Structured data, strong consistency, transactions, familiar SQL	RDS, Aurora, Azure SQL Database, Cloud SQL
NoSQL document and key-value	Flexible data, massive scale, fast simple lookups	DynamoDB, Cosmos DB, Firestore
Wide-column	Huge volumes of time-series and sensor data	Bigtable, Cassandra offerings
Data warehouse	Analytics across large datasets using SQL	Redshift, Synapse and Fabric, BigQuery
In-memory cache	Very fast temporary data to speed up applications	ElastiCache, Azure Cache for Redis, Memorystore

Choosing a database

- If you are unsure, start with a **managed relational database** such as PostgreSQL or MySQL. It suits most applications.
- Pick NoSQL when you have a clear need for its scale or flexible structure.
- Use a data warehouse for reporting and analysis rather than for running an app.

Database basics to get right

- Turn on **automated backups** and test a restore.
- Keep the database in a **private network**, not open to the whole internet (Chapter 7).
- Use a **strong password or managed identity** and limit who can connect.
- Choose a small size to start and scale up when you have real load.
- Delete test databases afterwards. They are one of the most common sources of surprise bills.

TIP

Serverless database options, which scale to zero when idle, are a good fit for learning projects, because they cost almost nothing when you are not using them.

Networking in the Cloud

Networking decides which resources can talk to each other and to the outside world. It looks intimidating, but only a few ideas matter at the start.

Virtual networks

A **virtual network** is your own private, isolated network in the cloud. On AWS and Google Cloud it is called a **VPC** (Virtual Private Cloud); on Azure it is a **Virtual Network (VNet)**. Within it you create **subnets**, which are smaller segments of the address range.

- A **public subnet** contains resources reachable from the internet, such as a web server or load balancer.
- A **private subnet** contains resources that should not be reachable from outside, such as databases.

Networks use **CIDR notation** for address ranges, such as 10.0.0.0/16. The number after the slash says how many addresses the range contains. Smaller numbers mean bigger ranges. Choose ranges that do not overlap with your office or other networks, since you may want to connect them later.

Controlling traffic: firewalls

Concept	AWS	Azure	Google Cloud
Instance-level firewall	Security group	Network security group (NSG)	VPC firewall rules
Subnet-level control	Network ACL	NSG on a subnet	Firewall policies

Rules say which source addresses may connect to which ports. Everything is typically **blocked by default**, and you open only what

you need. For example, port 443 for secure web traffic, 22 for SSH and 3389 for Windows remote desktop.

WARNING

Never open SSH (22), remote desktop (3389) or database ports to "anywhere" (0.0.0.0/0). Restrict them to your own IP address, or use a bastion host or the provider's secure connection service instead.

Public and private addresses

- Every VM has a **private IP** within the network.
- A **public IP** is optional, and makes the resource reachable from the internet. Fewer public IPs mean fewer risks.
- Resources in a private subnet that need to reach the internet (for updates) use a **NAT gateway** or similar. Note that NAT gateways cost money for as long as they exist.

Load balancers

A **load balancer** spreads incoming traffic across several servers, and stops sending traffic to those that fail health checks. This improves both capacity and reliability. All three clouds offer managed load balancers for web traffic and for lower-level network traffic.

DNS and domain names

DNS turns names such as `example.com` into IP addresses.

Managed DNS (Route 53, Azure DNS, Cloud DNS) lets you manage your records alongside your other resources. You can register a domain with a registrar or with the cloud provider.

Content delivery networks (CDNs)

A CDN keeps copies of your content in locations around the world, so users download from somewhere nearby. It speeds up sites, reduces load on your servers and can lower data transfer costs. Examples are CloudFront, Azure Front Door and CDN, and Cloud CDN.

Connecting to your own network

- **VPN** connections create an encrypted tunnel over the internet to your office or home network.
- **Dedicated connections** (such as AWS Direct Connect, Azure ExpressRoute and Google Cloud Interconnect) are private links for demanding workloads.
- **Peering** connects two virtual networks so they can talk privately.

Network troubleshooting checklist

1. Is the resource running and does it have the address you think?
2. Does the **firewall rule** allow the port from your address?
3. Is there a **route** to the internet, if one is needed?
4. Is the service itself listening on that port inside the machine?
5. Is a second firewall, such as one inside the operating system, blocking the traffic?

TIP

When something cannot connect, the cause is almost always a firewall rule, a missing public address or a wrong subnet. Check those three first.

Security and Identity

Security is not an extra on top of the cloud; it is part of every decision. The good news is that a few habits prevent the large majority of incidents.

The shared responsibility model

The provider and you each secure part of the system:

- **The provider** is responsible for security **of** the cloud: data centres, hardware, and the underlying platform.
- **You** are responsible for security **in** the cloud: your accounts, access permissions, data, configurations, operating systems you manage, and application code.

The more you use managed services, the more the provider handles. But you are always responsible for who has access and for how things are configured. Most cloud breaches are caused by misconfiguration and stolen credentials, not by failures of the provider.

Identity and access management (IAM)

IAM decides **who** can do **what** to **which** resources. The core ideas are the same on all three:

Concept	Meaning	Notes by provider
User	A person with a sign-in	AWS IAM or Identity Center users; Azure Entra ID users; Google accounts
Group	A collection of users that share permissions	Available on all three

Concept	Meaning	Notes by provider
Role	A set of permissions that can be assigned or temporarily assumed	AWS roles; Azure role assignments; Google IAM roles
Policy	A rule that grants or denies permissions	AWS policies; Azure RBAC and Policy; Google IAM policies
Service identity	An identity for an application, not a person	AWS roles for services; Azure managed identities; Google service accounts

The principle of least privilege

Give every person and application **only the permissions they need**, and no more. Start with narrow permissions and widen them if necessary. Avoid giving everyone administrator or owner rights "to make things work".

Essential security habits

1. **Enable multi-factor authentication** for every human user, especially administrators and the account owner.
2. **Protect the root or owner account.** Use it only for the few tasks that need it.
3. **Never put access keys in code** or commit them to public repositories. Attackers scan for them constantly.
4. **Prefer roles and managed identities** over long-lived keys. They issue temporary credentials automatically.
5. **Keep storage private** unless you have a clear reason otherwise.
6. **Restrict network access** with firewalls and private subnets (Chapter 7).
7. **Encrypt data** at rest and in transit. This is on by default for most services; keep it that way.

8. **Keep systems patched.** Use managed services or automated updates where possible.
9. **Turn on logging and alerts** so you can see who did what and when.
10. **Review permissions** regularly and remove accounts and keys that are no longer needed.

Secrets management

Passwords, API keys and certificates should live in a **secrets service** (AWS Secrets Manager, Azure Key Vault, Google Secret Manager), not in source code or configuration files. Applications then fetch them securely at run time.

Logging and monitoring

- **Audit logs** record every action taken in your account (CloudTrail, Azure Activity Log, Cloud Audit Logs). They are essential for investigating problems.
- **Monitoring** tools (CloudWatch, Azure Monitor, Cloud Monitoring) track performance and can alert you to unusual behaviour.
- **Security services** (such as GuardDuty, Microsoft Defender for Cloud and Security Command Center) look for threats and weak configuration.

If your account is compromised

1. **Change the password** and revoke or delete any exposed keys at once.
2. **Enable MFA** if it was not already on.
3. **Review audit logs** and your resource list for anything you did not create, such as unfamiliar VMs, which are often used to mine cryptocurrency.
4. **Delete the unauthorised resources** and check your billing.

5. **Contact the provider's support**, who can help and may be able to review charges.

WARNING

An access key pasted into a public code repository can be found and abused within minutes. If you ever expose one, treat it as compromised immediately, even if you delete it from the repository, because the history may be copied.

Understanding and Controlling Costs

The cloud's flexibility is also its biggest financial risk. A forgotten resource keeps billing. A mis-sized database can cost far more than expected. Learning to read and control costs is as important as learning to build.

What you are charged for

Cost driver	Details
Compute time	Billed while a VM, container or database is running, often per second
Storage	Charged per gigabyte per month, including disks of stopped VMs and snapshots
Data transfer out	Moving data into the cloud is usually free; moving it out to the internet or across regions is charged
Requests and operations	Per-request charges on storage, functions, queues and APIs
Managed services	Databases, load balancers and gateways are charged for as long as they exist
Support plans	Optional paid support tiers
Licences	Windows, SQL Server and some software add extra fees

Common causes of surprise bills

- **Idle resources** left running, such as forgotten VMs and databases.
- **Unattached disks and old snapshots** that remain after the VM is deleted.

- **Public IP addresses** that are reserved but not attached.
- **NAT gateways and load balancers**, which bill by the hour even with little traffic.
- **Data transfer** between regions or out to the internet.
- **Oversized instances** chosen "just in case".
- **Compromised accounts** used to mine cryptocurrency.
- **Leaving a free tier** by using a larger size or longer hours than the free allowance covers.

Set budgets and alerts first

Before you create anything, set up a budget with email alerts at a low amount. Each provider offers this for free:

Provider	Where to find it
AWS	Billing and Cost Management: Budgets (a "zero-spend" or small monthly budget works well when learning)
Azure	Cost Management and Billing: Budgets
Google Cloud	Billing: Budgets and alerts

WARNING

Budget alerts **notify** you; they usually do **not** stop spending automatically. Check your costs regularly, especially during your first weeks, and delete what you are not using.

Ways to pay less

- **Right-size.** Choose the smallest machine that does the job, and review it after real use.
- **Switch things off.** Stop development and test systems outside working hours.
- **Use serverless and autoscaling**, so you pay only for actual use.

- **Commit for a discount.** Savings plans, reserved instances and committed use discounts cut prices significantly in exchange for a one- or three-year commitment, once your usage is predictable.
- **Use spot or preemptible capacity** for flexible jobs that can be interrupted, at large discounts.
- **Use storage classes and lifecycle rules** to move old data to cheaper tiers.
- **Minimise data transfer**, and keep related services in the same region.
- **Delete the unused:** old snapshots, disks, test databases and addresses.

Tags, labels and cost reports

Tag every resource with a project and owner. Then use the provider's cost tools (AWS Cost Explorer, Azure Cost Analysis, Google Cloud Billing reports) to see exactly where money goes by service, project and day.

Estimating before you build

Each provider has a free **pricing calculator** (the AWS Pricing Calculator, the Azure pricing calculator and the Google Cloud pricing calculator). Use them to estimate monthly costs before you commit, and compare with your real bills afterwards.

A weekly cost-hygiene routine

1. Open the cost report and look for any unexpected change.
2. List running resources in every region you use.
3. Delete anything you no longer need.
4. Check that budget alerts are working and reaching you.

TIP

When you finish a learning project, delete the whole resource group (Azure), the project (Google Cloud) or the stack of resources (AWS). Removing the container removes everything inside it in one step.

Your First Projects

The fastest way to learn is to build something small. These two projects use inexpensive, widely available services and teach the key skills. Do them in a fresh account, with a budget alert set, and clean up when you are done.

Project 1: A static website

A static site is made of plain files (HTML, CSS, images) that need no server software. Hosting it from object storage is cheap, fast and a classic first project.

Step 1: Create a simple page

```
<!DOCTYPE html>
<html lang="en">
<head><title>My Cloud Site</title></head>
<body>
  <h1>Hello from the cloud</h1>
</body>
</html>
```

Save this as `index.html`.

Step 2: Host it on each cloud

Google Cloud (Cloud Storage):

```
gcloud storage buckets create gs://my-site-12345 \
  --location=us-central1
gcloud storage cp index.html gs://my-site-12345/
gcloud storage buckets update gs://my-site-12345 \
  --web-main-page-suffix=index.html
# Make objects readable by anyone (public site only)
gcloud storage buckets add-iam-policy-binding \
  gs://my-site-12345 --member=allUsers \
  --role=roles/storage.objectViewer
```

Azure (Storage static website):

```
az storage blob service-properties update \
  --account-name myuniquestore123 --static-website \
  --index-document index.html --auth-mode login
az storage blob upload --account-name myuniquestore123 \
  --container-name '$web' --name index.html \
  --file index.html --auth-mode login
az storage account show -n myuniquestore123 \
  --query "primaryEndpoints.web"
```

AWS (S3 static website):

1. Create a bucket with a unique name: `aws s3 mb s3://my-site-12345`.
2. Upload your file: `aws s3 cp index.html s3://my-site-12345/`.
3. In the S3 console, open the bucket's **Properties** tab and enable **static website hosting**, with `index.html` as the index document.
4. AWS blocks public access by default. For a public site, either relax the block and add a read-only bucket policy, or (preferably) put **CloudFront** in front of the private bucket.
5. Open the website endpoint shown in the console.

WARNING

Making a bucket public exposes everything in it. Use a bucket only for the public site, put no private files in it, and delete it when finished. Providers display clear warnings when you make storage public, so read them.

Project 2: A serverless function

A function responds to a request without any server for you to manage. Each provider offers a quick-start in the console that creates one in minutes.

1. **Create the function.** Open Lambda (AWS), Functions (Azure) or Cloud Run functions (Google Cloud) and choose "Create function".
2. **Pick a runtime**, such as Python or Node.js, and a template such as "HTTP trigger" or "Hello world".
3. **Enable a URL** that triggers it (a function URL, HTTP trigger or public endpoint).
4. **Edit the code** to return a custom message, and deploy.
5. **Call the URL** from your browser, then check the logs and the metrics that show the invocation.
6. **Delete it** when you have finished.

A minimal Python function looks like this in spirit (the exact signature differs by platform):

```
def handler(request):  
    return "Hello from a cloud function!"
```

Project ideas to try next

- Launch a small VM, install a web server and serve a page.
- Create a managed database and connect an app to it.
- Containerise a small app and deploy it to a serverless container service.
- Write a function that resizes an image whenever one is uploaded to a bucket.
- Set up a simple pipeline that deploys your site automatically when you push code to a repository.
- Recreate your project with Terraform so it can be rebuilt with one command.

Clean up

When you finish each project, delete what you created and confirm in the billing console that nothing is still running. Appendix A has a clean-up checklist.

TIP

Take notes as you build. A short log of what you created, which region and which commands you used makes cleaning up and repeating the project much easier.

Choosing a Cloud and Growing Your Skills

With the basics in place, you will want to decide which platform to focus on and how to keep learning.

How to choose

Situation	A natural fit
Your organisation already uses Microsoft 365, Windows Server or Active Directory	Azure
You want the widest range of services and the largest community and job market	AWS
Your work centres on data analytics, machine learning or Kubernetes	Google Cloud
You are a student or hobbyist learning the fundamentals	Any. Start with the one whose free offer and console you like
Your employer or client has already chosen	That one

In reality, many organisations use more than one cloud. Learning one deeply makes the others far easier to pick up, because the ideas match.

Multi-cloud: pros and cons

- **Advantages:** avoids reliance on one vendor, lets you use each provider's strengths, and can help with regional or regulatory needs.

- **Disadvantages:** more complexity, more skills to maintain, harder security management, and you often lose the benefits of provider-specific services.
- **For beginners:** learn one first. Use tools that work everywhere, such as containers and Terraform, to keep your options open.

A learning path

1. **The fundamentals:** what the cloud is, regions, billing and the shared responsibility model (this book).
2. **Linux and networking basics:** the command line, SSH, IP addresses, DNS and HTTP.
3. **Core services on one cloud:** compute, storage, databases, networking and identity.
4. **Containers:** Docker first, then a managed container service.
5. **Infrastructure as code:** Terraform or the provider's native tool.
6. **Automation and CI/CD:** deploying changes automatically and safely.
7. **Monitoring and reliability:** logs, metrics, alerts and backups.
8. **Specialise:** security, data engineering, machine learning, DevOps, or architecture.

Certifications

Certifications are not essential, but they give structure to your study and are recognised by employers. Each provider has an entry-level, fundamentals exam designed for beginners:

Provider	Entry-level certification (check current name)
AWS	AWS Certified Cloud Practitioner
Azure	Microsoft Azure Fundamentals (AZ-900)
Google Cloud	Cloud Digital Leader

After the fundamentals, each provider offers associate-level exams for administrators, developers and architects. Exam names and content change, so check the provider's certification pages.

Where to learn for free

- Each provider's **official documentation, tutorials and training portals** (AWS Skill Builder, Microsoft Learn and Google Cloud Skills Boost) offer free courses and hands-on labs.
- **Free tiers and credits**, used carefully, give you real practice.
- **Community forums, user groups and videos** help when you are stuck.
- **Open-source projects** and your own side projects teach more than any course.

Habits of successful cloud learners

- **Build things.** Reading is not enough.
- **Break things safely**, in a sandbox account with a budget.
- **Read error messages** carefully; they usually tell you exactly what is wrong.
- **Read the documentation** for current, accurate details.
- **Clean up every time.**
- **Keep security first** from day one.

TIP

Cloud skills age quickly, but fundamentals do not. Invest most of your time in the concepts: networking, identity, storage, cost and reliability. They carry across every platform and every job.

Appendix A: Checklists and Commands

New account checklist

1. **Sign up** with an email address you control.
2. **Enable multi-factor authentication** on the main account.
3. **Create a daily-use user** and stop using the root or owner account.
4. **Set a budget alert** with a low limit.
5. **Choose your region** and record it.
6. **Confirm free-tier terms** on the provider's current pages.
7. **Install the CLI** or use the cloud shell.

Clean-up checklist

1. **Check you are in the right account, subscription or project.**
2. **Delete VMs, containers and functions.**
3. **Delete disks and snapshots** that were left behind.
4. **Delete databases** and backups you no longer need.
5. **Release public IP addresses** and delete load balancers and NAT gateways.
6. **Empty and delete buckets.**
7. **Delete the resource group or project** if the work is finished.
8. **Check the billing report** the next day.

Command quick reference

Command	What it does
<code>aws configure</code>	Set up AWS CLI credentials and defaults
<code>aws sts get-caller-identity</code>	Show which AWS identity you are using
<code>aws ec2 describe-instances</code>	List EC2 instances
<code>aws s3 ls</code>	List S3 buckets
<code>az login</code>	Sign in to Azure
<code>az account show</code>	Show the current subscription
<code>az group list</code>	List resource groups
<code>az vm list -o table</code>	List virtual machines
<code>az group delete --name NAME</code>	Delete a resource group and its contents
<code>gcloud init</code>	Initialise the Google Cloud CLI
<code>gcloud config list</code>	Show the active project and settings
<code>gcloud compute instances list</code>	List Compute Engine VMs
<code>gcloud storage ls</code>	List Cloud Storage buckets

Appendix B: Glossary

Autoscaling. Automatically adding or removing capacity as demand changes.

Availability zone. One or more separate data centres within a region, with independent power and networking.

Bucket. A container for objects in object storage.

CDN. Content delivery network: servers around the world that deliver content from a location near the user.

CIDR. A notation such as 10.0.0.0/16 for describing a range of IP addresses.

CLI. Command-line interface: a tool for managing cloud resources by typing commands.

Container. A package that bundles an application with what it needs to run anywhere.

Egress. Data leaving a cloud network, which is usually charged.

Function (serverless). A small piece of code run on demand without managing servers.

IaaS, PaaS, SaaS. Service models: infrastructure, platform and software as a service.

IAM. Identity and access management: controlling who can do what.

Infrastructure as code. Defining infrastructure in text files so it can be created repeatably.

Kubernetes. A system for running and managing many containers across many machines.

Least privilege. Giving only the permissions that are needed.

Load balancer. A service that spreads traffic across several servers.

MFA. Multi-factor authentication: proving identity with more than a password.

Object storage. Storage for files accessed over the web, organised into buckets.

Region. A geographic area containing cloud data centres.

Resource. Any item you create in the cloud, such as a VM or a database.

Resource group. An Azure container that groups related resources.

Role. A set of permissions that can be assigned to a user or service.

Serverless. Services where the provider manages servers and you pay per use.

Shared responsibility. The split of security duties between the provider and the customer.

Subnet. A segment of a virtual network.

Tag / label. A key-value note attached to a resource for organising and billing.

VPC / VNet. A private virtual network in the cloud.

Virtual machine (VM). A software-based computer rented from the provider.