

universidade de aveiro



deti

departamento de eletrónica,  
telecomunicações e informática

# Infrastructure as Code

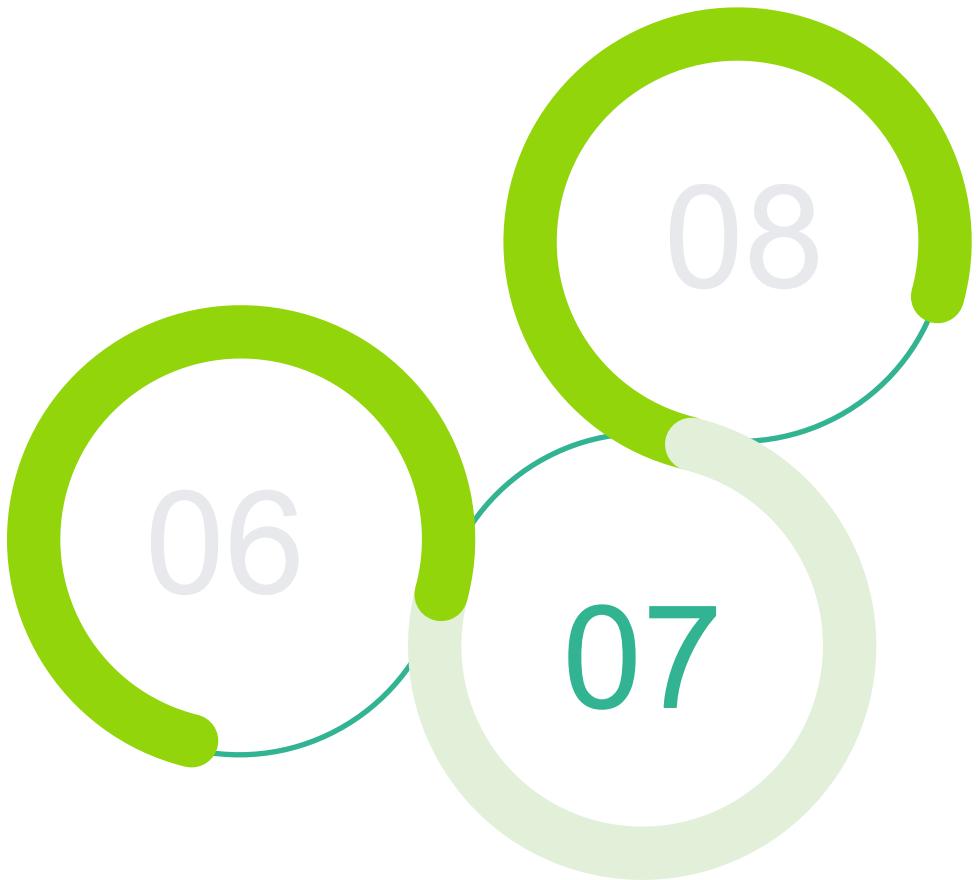
Software Engineering

2025/2026



# Outline

- Deployment History
  - Pre-cloud Era
  - Cloud Era
  - DevOps Recap
- Infrastructure as Code (IaC)
  - What is IaC?
  - IaC Tools
- Terraform 101
- Scalability & Availability



# Deployment History

# CAPEX vs OPEX

## ➤ Capital Expenditure

- Upfront investment in physical infrastructure
- Large initial cost
- Fixed assets
- Long-term expenses
- Ownership of infrastructure

## ➤ Operational Expenditure

- Pay-as-you-go model
- No depreciation
- Operational expenses
- Flexible scaling costs
- No ownership required



**CAPEX**



**OPEX**

# Pets vs Cattle servers

## ➤ Pets

- **Individually named and cared for:** Each server has a unique name (like "db-server-prod-01" or "WebServerJohn")
- **Lovingly maintained:** You diagnose and manually fix each problem
- **Irreplaceable:** Server failure causes crisis and significant downtime
- **Manually configured:** Hand-crafted with custom settings and accumulated tweaks
- **Long-lived:** Run for years, becoming difficult to replicate
- **You know when they're sick:** Problems trigger immediate all-hands response



# Pets vs Cattle servers

## ➤ Cattle

- **Numbered, not named:** Servers are "web-server-001", "web-server-002"
- **Disposable:** Terminate and replace rather than fix problems
- **Automatically provisioned:** Created from templates/code in minutes
- **Ephemeral:** Expected to be short-lived for easy updates
- **Scalable:** Add or remove servers automatically as needed
- **Self-healing:** System auto-replaces failures without human intervention



# Pre-Cloud Era

## ➤ On-Premise Model

- Capex
- Pet Servers
- Buy Physical Servers
- High Initial Cost
- Difficult to Scale
- Manual Configuration

amazon Deliver to Portugal

All Computer Servers

1-24 of over 10,000 results for "Computer Servers"

**Brands**

- Dell
- HP
- Supermicro
- Lenovo
- Cisco
- HEWLETT PACKARD
- Intel

[See more](#)

**Customer Reviews**

★★★★★ & Up

**Condition**

- New
- Renewed
- Used

**Price**

\$1 – \$18,600+

Up to \$100  
\$100 to \$250  
\$250 to \$350  
\$350 & above

**Deals & Discounts**

All Discounts  
Today's Deals

**Seller**

A-Tech Components

**Results**

Check each product page for other buying options.

**Hewlett Packard Enterprise ProLiant MicroServer Gen11 Tower Server, Intel Xeon E-2434 Processor, 32GB Memory, 4TB HDD Storage, External 180W US Power Supply (HPE Smart Choice P74440-005)**

4.4 ★★★★☆ (122)

[See options](#)

No featured offers available

\$1,179.00 (12 used & new offers)

**Dell PowerEdge R730xd Server 12-Bay LFF 2.60Ghz 28-Core 192GB RAM 14x Caddies (Renewed)**

3.4 ★★★★☆ (6)

**\$559<sup>00</sup>**

Delivery Thu, Nov 6  
Ships to Portugal

[See options](#)

**GMKtec Mini PC Workstation, Intel Core i9 13900HK(14C/20T) up to 5.4GHz, Mini Computer 32GB DDR5 RAM 1TB SSD, 8X USB Ports/COM/HDMI/DP Office Business**

Options: 5 sizes

4.4 ★★★★☆ (279)

300+ bought in past month

**\$539<sup>99</sup>** List Price: \$718.99  
5% off promotion available

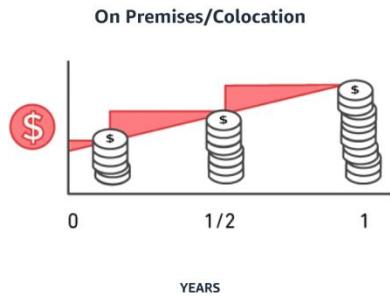
Delivery Mon, Nov 3  
Ships to Portugal

[See options](#)

# Cloud Era

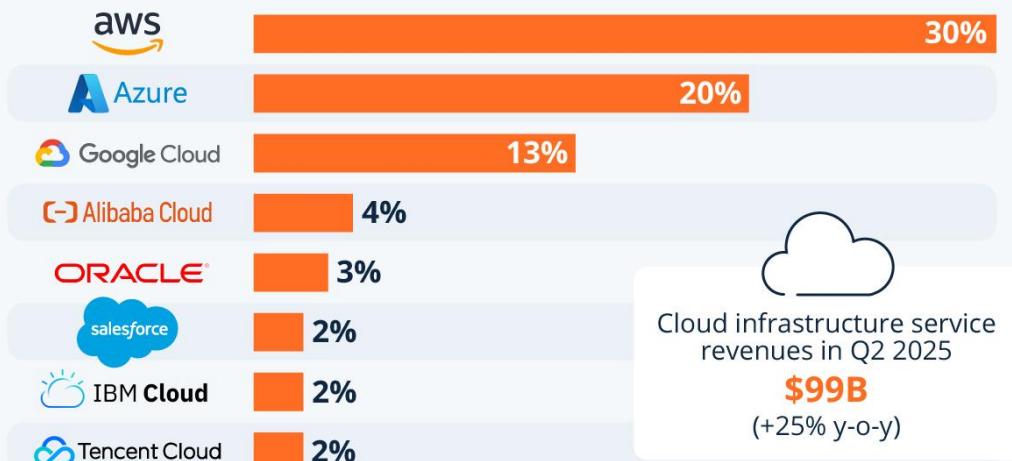
## ➤ Cloud Model

- Opex
- Cattle Servers
- Pay-per-usage
- Provisioning on-the-fly
- Scalability on-demand
- Easier for startups



## The Big Three Stay Ahead in Ever-Growing Cloud Market

Worldwide market share of leading cloud infrastructure service providers in Q2 2025\*



\* Includes platform as a service (PaaS) and infrastructure as a service (IaaS) as well as hosted private cloud services

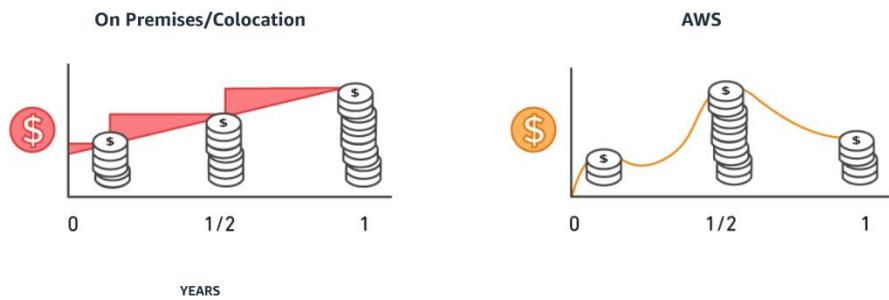
Source: Synergy Research Group



# Cloud Era

## ➤ Cloud Model

- Opex
- Cattle Servers
- Pay-per-usage
- Provisioning on-the-fly
- Scalability on-demand
- Easier for startups



[Customer Stories / Financial Services](#)

2020



## Capital One Completes Migration from Data Centers to AWS, Becomes First US Bank to Announce Going All In on the Cloud

Capital One exited eight on-premises data centers by migrating to AWS, transforming the customer experience in the process.

[Overview](#) | [Opportunity](#) | [Solution](#) | [Outcome](#) | [AWS Services Used](#)

<b>8</b> closed on-premises data centers, migrating completely to the cloud	<b>11,000</b> member technology team during 8-year migration journey	<b>70%</b> better disaster recovery time in tests	<b>50%</b> reduction in number of transaction errors and reduced critical incident resolution time	<b>3 months to minutes</b> for the average development environment build time
--	---	--	---	--

# Not All Days Are Sunny Days

Computing > Internet

• LIVE Is AWS down? Service seems to be back as Amazon disputes outage and users disagree

AWS is disputing the outage, but users disagree

News By Jason England, Dave LeClair last updated 1 hour ago



When you purchase through links on our site, we may earn an affiliate commission. [Here's how it works.](#)



Microsoft Azure (Opens in a new tab)  owndetector! by Ookla®

US For Business About us Search

## Azure

User reports indicate problems at Microsoft Azure

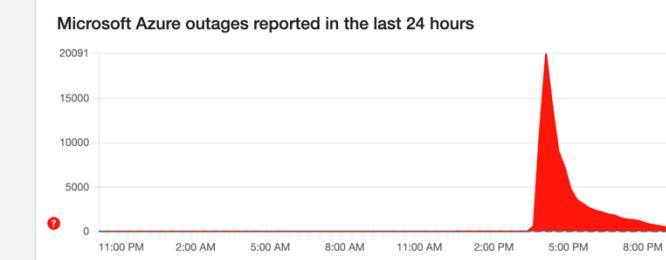
Microsoft Azure is a cloud computing platform operated by Microsoft. Azure offers both 'platform as a service' (PaaS) and 'infrastructure as a service' cloud solutions.

I have a problem with Microsoft Azure

Select the option you are having issues with and help provide feedback to the service.

Website Server Connection Domains Something else...

Microsoft Azure outages reported in the last 24 hours



Having problems? Complain here:  
Via X  
Via Facebook  
Check the official [status page](#)

Check the status of Microsoft Azure in other countries



Posts by @azuresupport

Downtector 75,911 followers

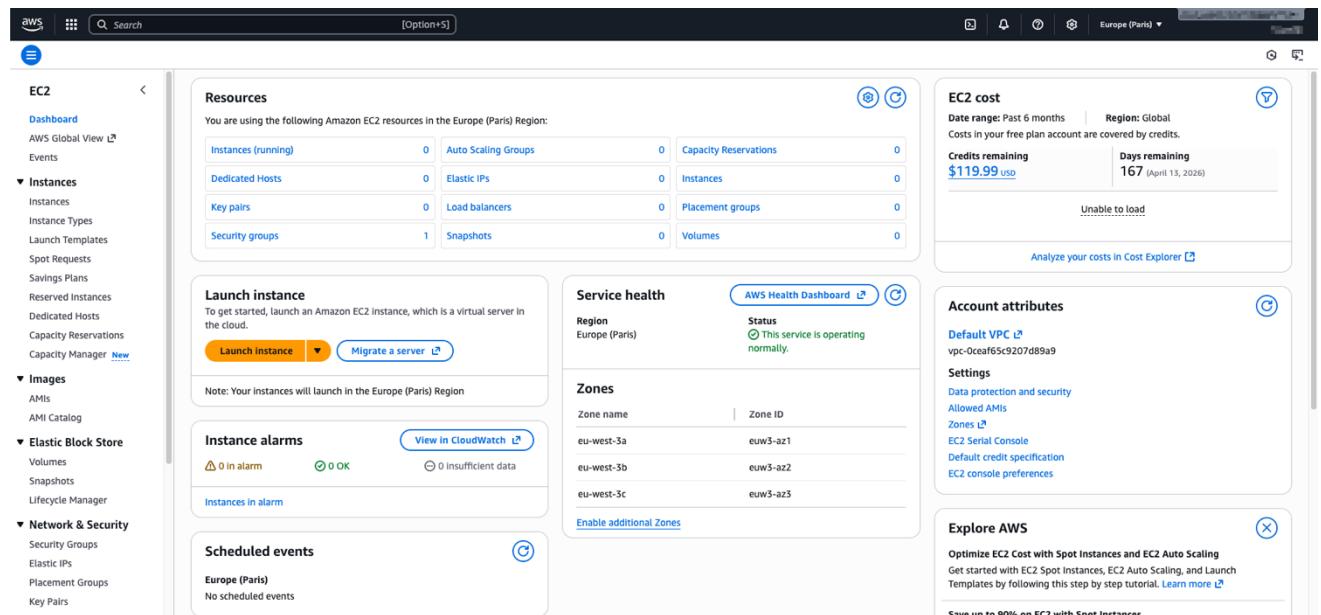
Follow Page Share



# Configuration Problem

## ➤ Manual Configuration

- Even when using cloud solutions, it is needed **manual configuration**.
- **Snowflake servers**, that diverge over time and become hard to replicate
- No **versioning control** making impossible to rollback



# Configuration Problem

## ➤ Manual Configuration

- Even when using cloud solutions, it is needed **manual configuration**.
- **Snowflake servers**, that diverge over time and become hard to replicate
- No **versioning control** making impossible to rollback

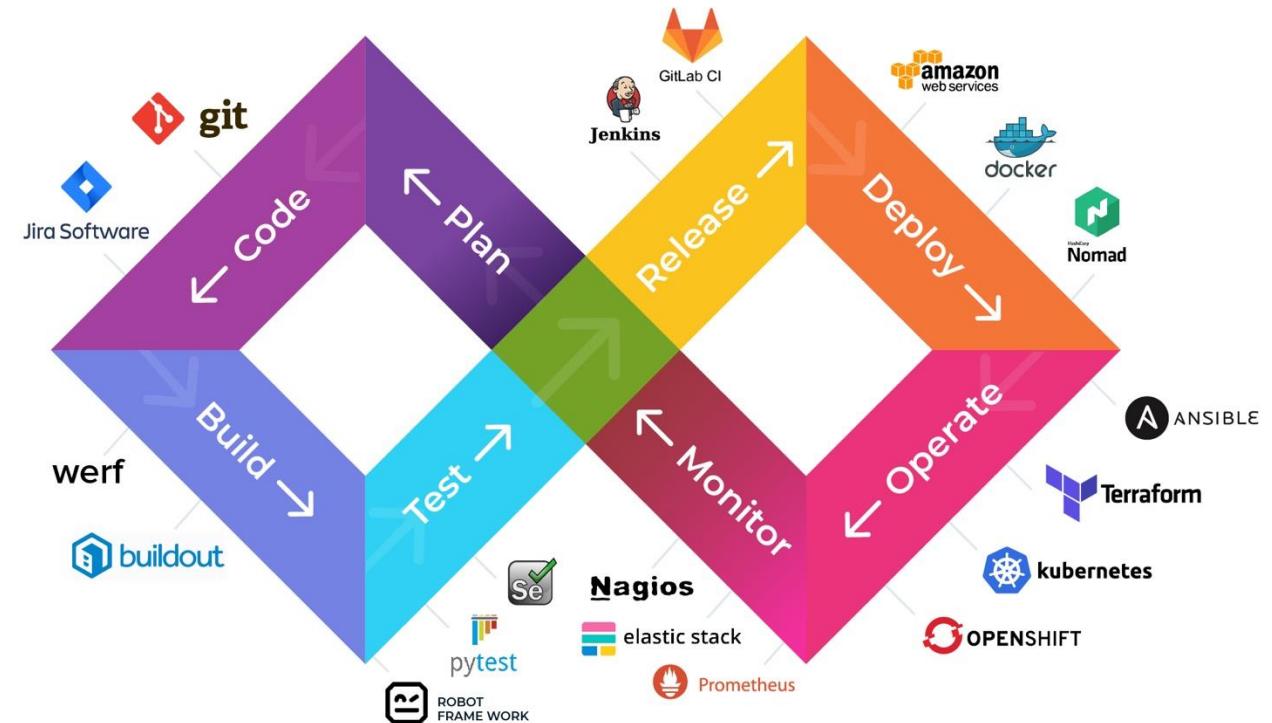


## Solution: Infrastructure as Code

# DevOps Recap

## ➤ Movement

- Culture, automation, measurement and sharing (CAMS)
- Join the Developer Team with the Operator Team
- Combat the “*It works on my machine syndrome!*”
- Enhance Software delivery



# DevOps Recap

## ➤ Operate Tools

- Running and maintaining applications in production environments
- Ensuring system reliability, performance, and availability
- Managing infrastructure and responding to incidents



# DevOps Recap

Table P-1. DORA metrics performance from the [2024 State of DevOps Report](#)

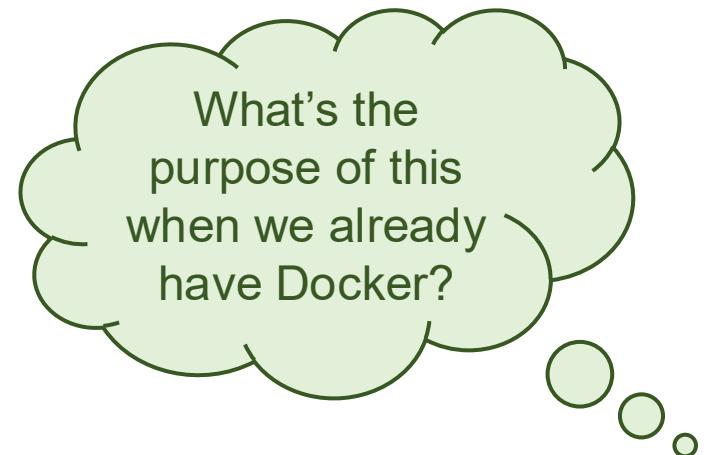
Metric	Description	Elite vs. low performers
Deployment frequency	How often you deploy to production	182× more often
Lead time	How long it takes a change to go from committed to deployed	127× faster
Change failure rate	How often deployments cause failures that need immediate remediation	8× lower
Recovery time	How long it takes to recover from a failed deployment	2,293× faster

# Infrastructure as Code

IaC

# Overview

- Infrastructure as Code (IaC) is the idea of writing and executing code to define, deploy, update and destroy your infrastructure
  - Infrastructure defined as text files
  - Version Control (Git)
  - Declarative
  - Idempotent
  - Automatic and Reproducible



# IaC Tools: Docker

- **Docker is about running your app:**
  - It creates containers: self-contained environments that hold your app and its dependencies.
  - It ensures the same environment on any host.
  - It's great for build → ship → run consistency.
- **Example:**
  - You can run a web app in a Docker container with its specific version of Python, Nginx, and libraries, without caring about the host OS.

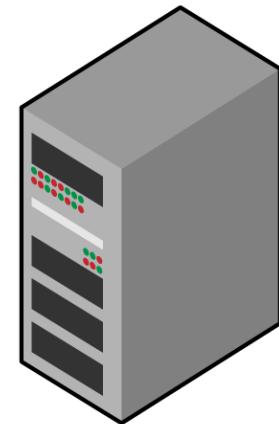
# IaC Tools: Ad-hoc scripts

## ➤ The most straightforward way

- Writing a simple script in your favorite scripting language (Bash, Python) to define each step to execute
- The main advantage is you can use your preferred language
- The main disadvantage is you can use your preferred language, provoking inconsistency
- More ad-hoc scripts generally means more spaghetti code to maintain.

```
apt-get update  
apt-get install \  
    -y \  
    php \  
    apache 2  
  
git clone \  
    github.com/foo/bar \  
    /var/www/html/app  
  
service apache2 start
```

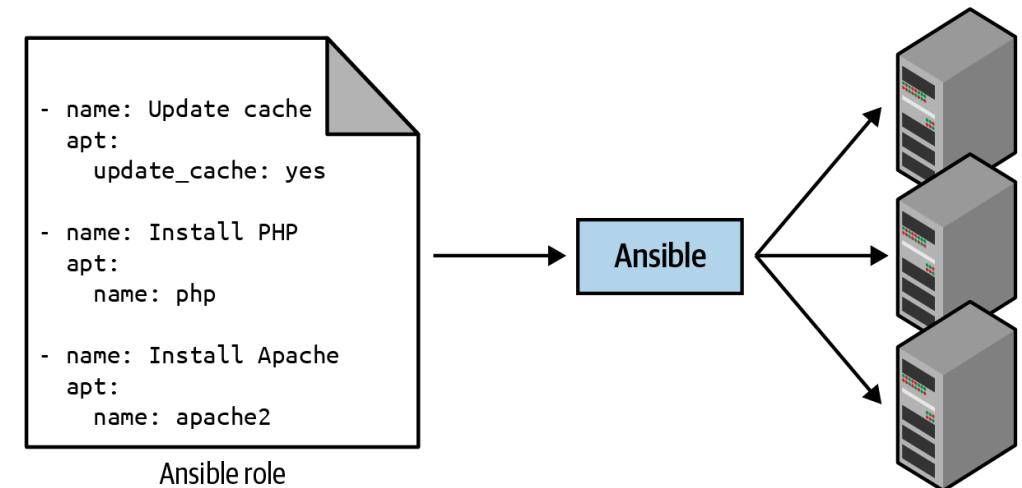
Ad hoc script



# IaC Tools: Configuration management tools

## ➤ The most common tools: Chef, Puppet, Ansible

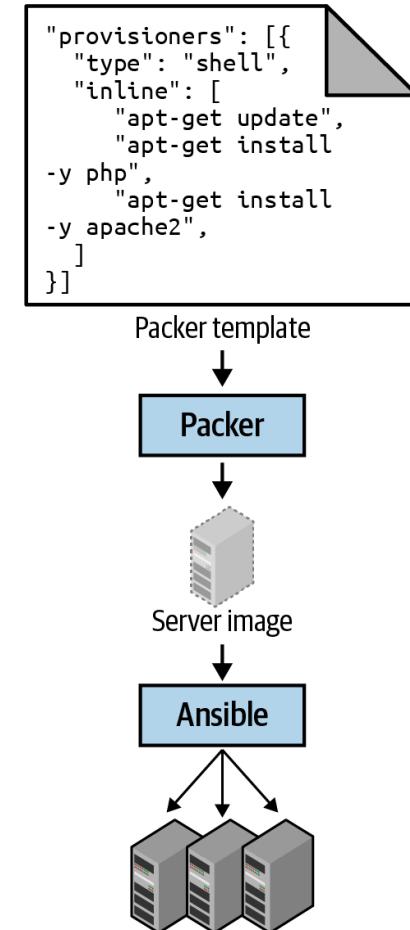
- Designed to install and manage software on existing servers
- Enforces consistent coding conventions including documentation, file layout, and secrets management
- The main advantage is **idempotence** - code that works correctly no matter how many times you run it
- Built for distribution - designed to manage large numbers of remote servers in parallel
- The main disadvantage is still requires managing the servers themselves



# IaC Tools: Server Templating Tools

## ➤ The most common tools: Docker, Packer, Vagrant

- Create an image that captures a fully self-contained snapshot of the OS, software, and files
- Two main types: Virtual Machines (VMware, VirtualBox) and Containers (Docker, rkt)
- The main advantage enables immutable infrastructure - servers never change after deployment
- The main disadvantage requires orchestration tools to manage the images at scale



# IaC Tools: Orchestration Tools

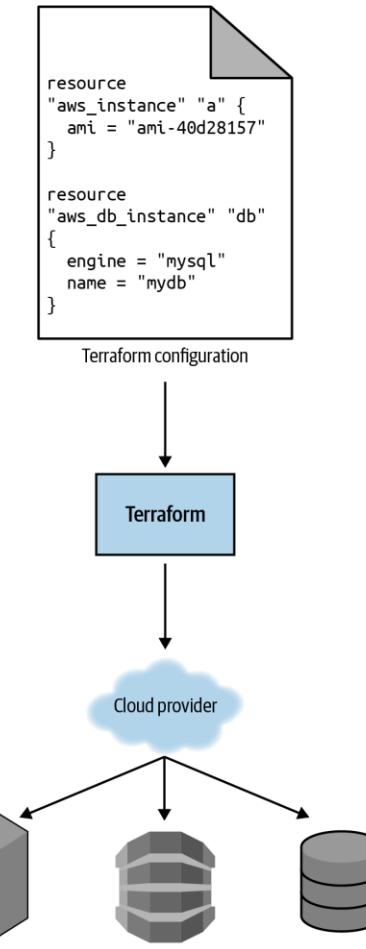
## ➤ The most common tools: Kubernetes, Docker Swarm

- Manage deployment of VMs and containers making efficient use of hardware
- Handle rolling updates, blue-green deployments, and canary deployments
- Provide auto-healing by monitoring health and replacing unhealthy instances
- Enable auto-scaling in response to load
- Distribute traffic through load balancing
- Facilitate service discovery so containers can communicate

```
deployment.yaml  
YAML  
  
apiVersion: apps/v1  
  
# Use a Deployment to deploy multiple replicas of your Docker  
# container(s) and to declaratively roll out updates to them  
kind: Deployment  
  
# Metadata about this Deployment, including its name  
metadata:  
  name: example-app  
  
# The specification that configures this Deployment  
spec:  
  # This tells the Deployment how to find your container(s)  
  selector:  
    matchLabels:  
      app: example-app  
  
  # This tells the Deployment to run three replicas of your  
  # Docker container(s)  
  replicas: 3  
  
  # Specifies how to update the Deployment. Here, we  
  # configure a rolling update.  
  strategy:  
    rollingUpdate:  
      maxSurge: 3  
      maxUnavailable: 0  
    type: RollingUpdate  
  
  # This is the template for what container(s) to deploy  
  template:  
    # The metadata for these container(s), including labels  
    metadata:  
      labels:  
        app: example-app  
  
    # The specification for your container(s)  
    spec:  
      containers:  
        # Run Apache listening on port 80  
        - name: example-app  
          image: httpd:2.4.39  
          ports:  
            - containerPort: 80
```

# IaC Tools: Provisioning Tools

- **The most common tools: Terraform, OpenTofu, Pulumi**
  - Responsible for creating the servers and infrastructure themselves
  - Create not only servers but databases, load balancers, networks, firewall rules, SSL certificates
  - Define infrastructure resources declaratively as code
  - Work across multiple cloud providers
  - Often used in combination with other IaC tools (config management, templating, orchestration)



# Benefits of IaC

- Entire deployment process automated, developers can deploy without bottlenecks
- Automated deployments are faster and more consistent than manual processes
- Infrastructure state is in source files everyone can read, not locked in one person's head

High-performing IT organizations report experiencing:

200x

200x more frequent deployments

24x

24x faster recovery from failures

3x

3x lower change failure rate

2,555x

2,555x shorter lead times

State of DevOps Report 2016

# Benefits of IaC

- Complete infrastructure history captured in commit logs for debugging and rollbacks
- Every change can go through code review, automated tests, and static analysis
- Package infrastructure into reusable modules instead of building from scratch each time
- Eliminates repetitive manual work, letting computers handle automation and developers focus on coding

High-performing IT organizations report experiencing:

**200x**

200x more frequent deployments

**24x**

24x faster recovery from failures

**3x**

3x lower change failure rate

**2,555x**

2,555x shorter lead times

State of DevOps Report 2016

# Terraform 101

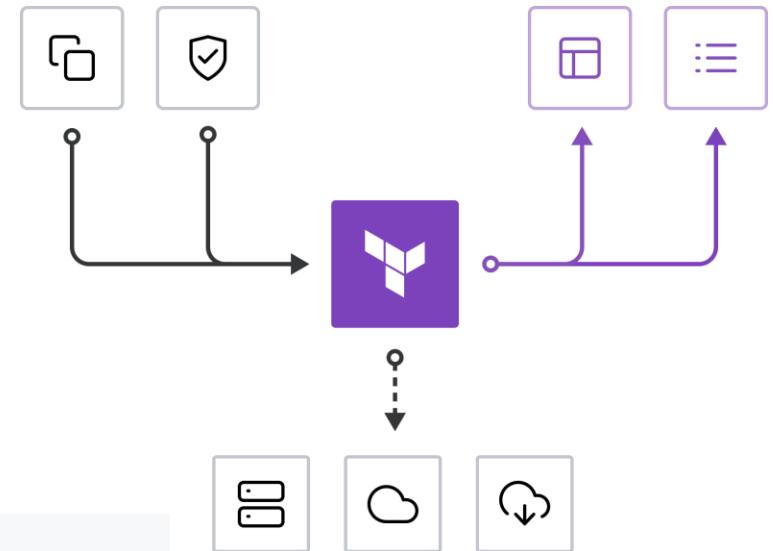
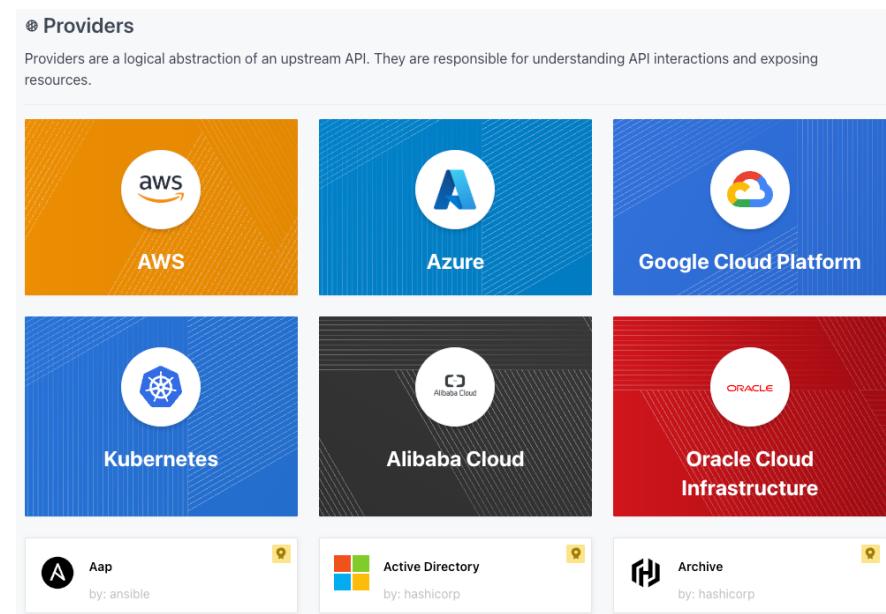
# Terraform

## ➤ What is Terraform?

- "Terraform is a tool for building, changing, and versioning infrastructure safely and efficiently."

## ➤ Main Features

- Open Source
- Declarative
- Cloud-agnostic
- Multiple Provider Support
- State Management
- Planned Execution



# Terraform: Declarative Infrastructure

## ➤ HCL (HashiCorp Configuration Language)

- Describe the desired state, Terraform handles the "how"
- Resources are the building blocks of infrastructure
- Terraform determines the order of operations automatically

main.tf

HCL

```
resource "aws_instance" "web_server" {
    ami           = "ami-0fb653ca2d3203ac1"
    instance_type = "t2.micro"

    tags = {
        Name = "MyWebServer"
    }
}
```

# Terraform: State Management

## ➤ Terraform tracks infrastructure state

- State file stores current infrastructure configuration
- Enables Terraform to detect what needs to be changed
- Can be stored locally or remotely (S3, Terraform Cloud)
- Allows collaboration across teams

backend.tf HCL

```
terraform {
  backend "s3" {
    bucket = "my-terraform-state"
    key    = "prod/terraform.tfstate"
    region = "us-east-1"
  }
}
```

# Terraform: Variables

## ➤ Make configurations reusable and flexible

- Variables allow parameterization of configurations
- Support different types: string, number, list, map, object
- Can have default values and validation rules
- Values can be provided via CLI, files, or environment variables

variables.tf

HCL

```
variable "instance_type" {  
  description = "EC2 instance type"  
  type        = string  
  default     = "t2.micro"  
}
```

main.tf

HCL

```
resource "aws_instance" "app" {  
  ami          = "ami-0fb653ca2d3203ac1"  
  instance_type = var.instance_type  
}
```

# Terraform: Output

## ➤ Expose information from your infrastructure

- Outputs display values after terraform apply
- Can be used by other Terraform configurations
- Useful for passing data between modules
- Help document important infrastructure details

```
outputs.tf HCL
_____
output "instance_ip" {
  description = "Public IP of the instance"
  value       = aws_instance.web.public_ip
}

output "instance_id" {
  description = "ID of the EC2 instance"
  value       = aws_instance.web.id
}
```

# Terraform: Resource Dependencies

## ➤ Automatic and explicit dependency management

- Terraform automatically detects dependencies
- Can explicitly define dependencies when needed
- Creates resources in correct order
- Handles parallel resource creation when possible

```
main.tf                                HCL

resource "aws_security_group" "web_sg" {
    name = "web-security-group"

    ingress {
        from_port   = 80
        to_port     = 80
        protocol    = "tcp"
        cidr_blocks = ["0.0.0.0/0"]
    }
}

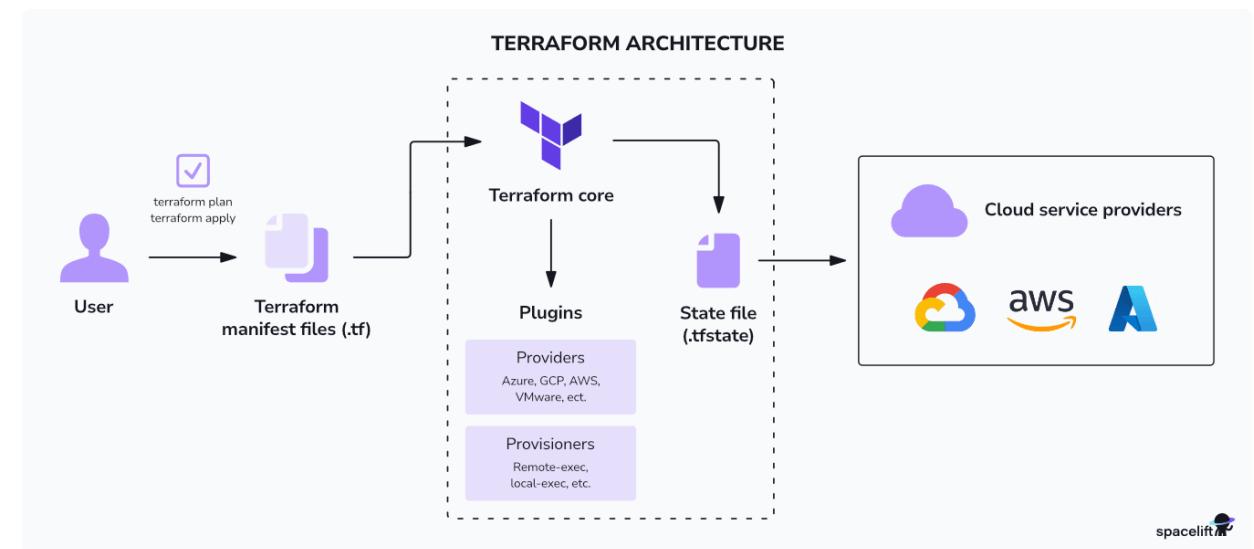
resource "aws_instance" "web" {
    ami           = "ami-0fb653ca2d3203ac1"
    instance_type = "t2.micro"
    security_groups = [aws_security_group.web_sg.name]

    depends_on = [aws_security_group.web_sg]
}
```

# Terraform: Workflow

## ➤ Three Steps when using Terraform

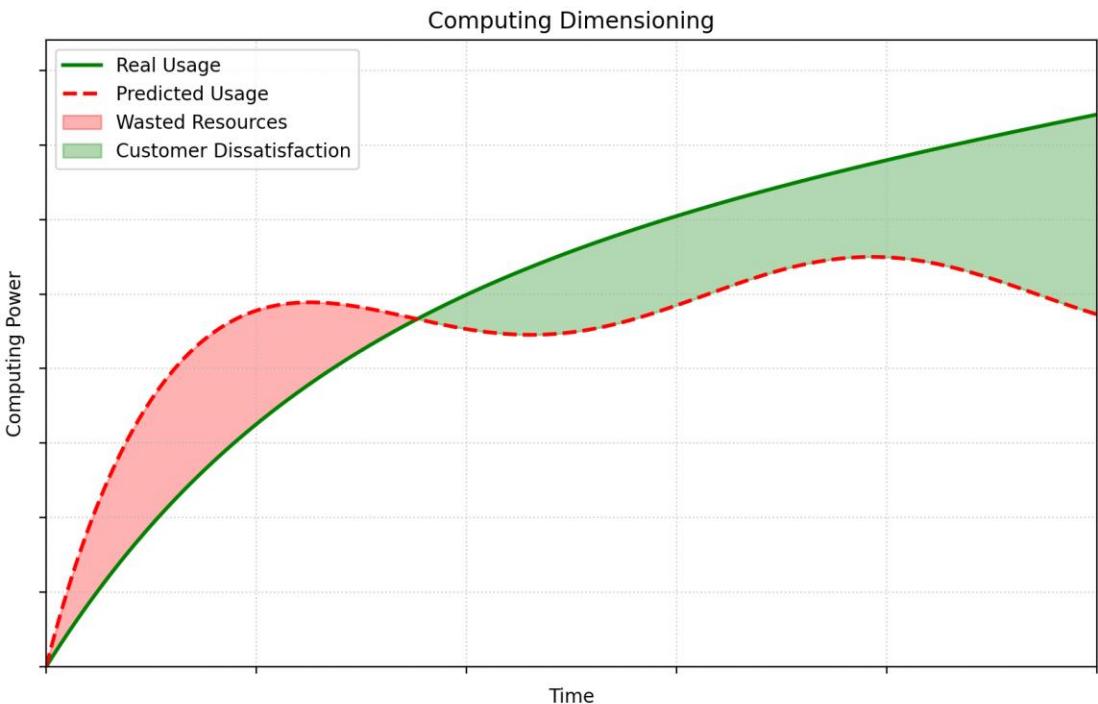
- **Write** - Author infrastructure as code.
- **Plan** - Preview changes before applying.
- **Apply** - Provision reproducible infrastructure.



# Scalability & Availability

# Overview

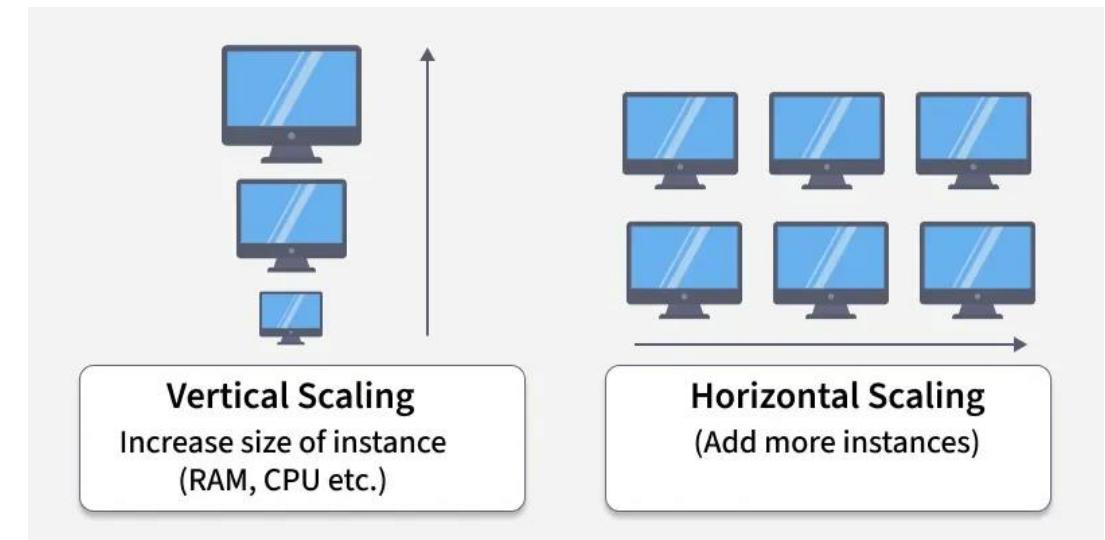
- **Modern applications need to handle growth and stay online**
  - Cloud infrastructure provides flexible and scalable options
  - Users expect systems to be always available and responsive
  - Traffic patterns can be unpredictable and change significantly
  - Systems must handle both normal operations and unexpected failures



# Scalability

Scalability is the ability of a system to handle increasing workload

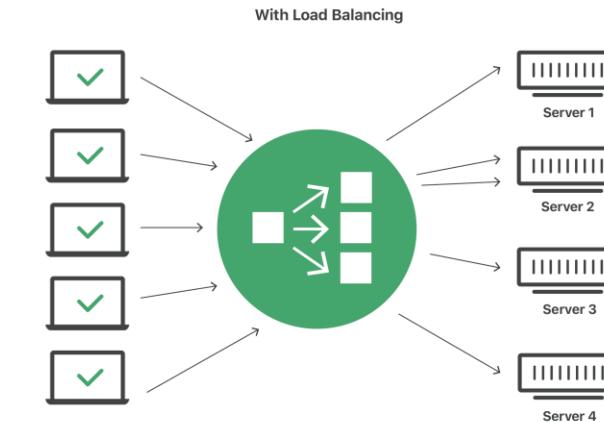
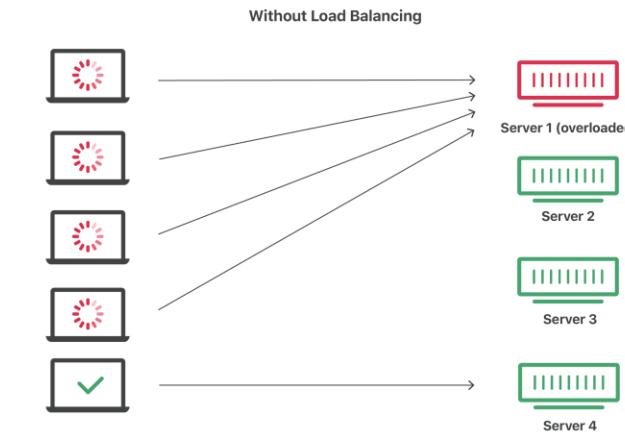
- Vertical scaling: adding more CPU, RAM, or storage to existing servers
- Horizontal scaling: adding more servers to distribute the load
- Auto-scaling adjusts resources based on demand
- Scalability =  $(\text{Maximum Load} / \text{Initial Load})$  or  $(\text{Performance at Scale} / \text{Baseline Performance})$



# Load Balancing

Load balancers distribute incoming traffic across multiple servers. This prevents any single server from being overwhelmed and improves overall performance.

- Distributes requests evenly across available servers
- Health checks detect and remove failing servers from rotation
- Load per server = Total Load / Number of Active Servers
- Supports different methods: round-robin, least connections, IP hash



# Availability

Availability measures how often your system is up and running.

- 99.9% availability = 8.76 hours downtime per year
- 99.99% availability = 52 minutes downtime per year
- Availability (%) =  $(\text{Uptime} / (\text{Uptime} + \text{Downtime})) \times 100$
- Redundancy removes single points of failure



# Practice Guide

# Practical Guide

- In this lab:
  - Learn from this tutorial (Use Terraform with Docker provider):
    - <https://developer.hashicorp.com/terraform/tutorials/docker-get-started>
  
- In your project:
  - You should change your deployment from docker compose into terraform scripts, only your frontend, backend and database.
  - They should be integrated in your CI-CD pipeline
  - You need to make sure that there are **two replicas** of your frontend and backend
  - If one of the services fail the application must be fault tolerant
    - TIP: You can use nginx as load balancer to achieve High Availability

# Bibliography

