What is infrastructure as code, and why use it?

1 November 2026 · 5 min read · blogs.iictp.org/what-is-infrastructure-as-code

Infrastructure as code (IaC) means describing servers, networks and cloud services in files that a tool applies, instead of building them by hand in a console. The files are versioned and reviewed like application code, so the environment can be recreated, compared and changed safely.

TL;DR The short version

- Infrastructure as code describes servers, networks and cloud services in versioned files that a tool applies.
- It makes environments repeatable, so staging can match production, and shows drift when someone changes things by hand.
- Change safely with a reviewed plan applied from a pipeline and policy checks, starting with one non-critical environment.

Read online, listen to it, or change the text size: https://blogs.iictp.org/what-is-infrastructure-as-code

What is the difference between declarative and imperative IaC?

A declarative definition states the end result you want, and the tool works out the steps. An imperative script lists the steps to run. Most modern tools are declarative, because re-running them is safe: they change only what differs from the desired state.

What are the main benefits?

Environments become repeatable, so staging can match production. Every change is recorded in version control, reviewed before it is applied, and can be rolled back. It also shortens recovery, because a lost environment can be rebuilt from the files.

What is drift, and how do you handle it?

Drift is the gap that opens when someone changes a resource by hand and the files no longer describe reality. Detect it by comparing the real state with the code on a schedule, and then either bring the code up to date or put the resource back. Restricting console access reduces it.

How should state and secrets be handled?

Many tools keep a state file recording what they created. Store it in a shared, access-controlled location with locking, since it may contain sensitive values. Keep secrets in a secret manager and reference them, rather than writing them into the files.

What is a safe change process?

Run a plan or preview that shows what will change, have a person review it, apply it from a pipeline rather than a laptop, and add policy checks that block risky changes such as public storage. Small, frequent changes are easier to review than large ones.

Where do you start?

Pick one non-critical environment and define it in code, then expand. Begin by importing existing resources rather than rebuilding them, and agree naming and module conventions early so teams do not diverge.

> Related credential: Cloud Architecture Practitioner. Explore the exam facts and syllabus.

Sources and further reading

1. HashiCorp. Terraform documentation. developer.hashicorp.com/terraform.
2. Morris, K. Infrastructure as Code. O’Reilly, 2nd ed. 2020.

Published by IICTP, an independent technical credential body, on 1 November 2026. The current version of this post is at https://blogs.iictp.org/what-is-infrastructure-as-code. More guides: blogs.iictp.org.