# **What is Infrastructure as Code?**

### **What is Infrastructure as Code?**

To understand the Infrastructure as Code concept better, I explain how DevOps tasks were done

* before automation and
* after automation.

#### **► DevOps tasks BEFORE automation**

Before automation you were doing everything by hand 🙌🏼:

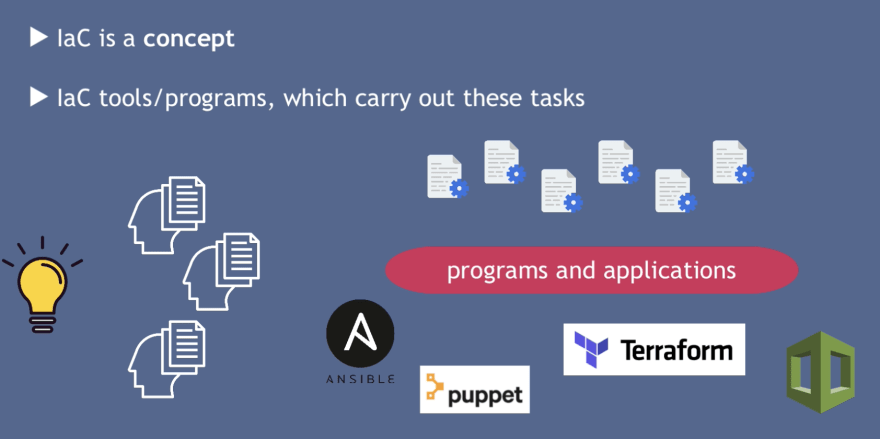
* setup servers
* configure networking
* install and configure software
* etc.

In addition to that all the tasks needed to be done for multiple environments like DEV, TEST and PROD environments.



Now, Infrastructure as Code is a way to automate all these DevOps tasks end to end instead of doing it manually.   
All the knowledge and expertise of system administrators or DevOps engineers are packed into programs and applications that carry out those tasks.

So, Infrastructure as Code or IaC is a concept and there are Infrastructure as Code tools, like Ansible, Puppet, Terraform or Cloudformation etc that you can use for different tasks.



Infrastructure as Code (IaC) is the practice of managing and provisioning computing infrastructure through machine-readable scripts, rather than manual hardware configuration or interactive configuration tools. It allows you to define and manage infrastructure (like servers, networks, databases, etc.) using code, enabling automation, version control, and consistency across environments.

## **IaC tools**

Server automation and configuration management tools can often be used to achieve IaC. There are also solutions specifically for IaC.

#### **These are some popular choices:**

* [Terraform](https://www.redhat.com/en/topics/automation/ansible-vs-terraform)
* [Puppet](https://www.redhat.com/en/topics/automation/ansible-vs-puppet)
* [Red Hat® Ansible® Automation Platform](https://www.redhat.com/en/technologies/management/ansible2)
* [Saltstack](https://www.redhat.com/en/topics/automation/ansible-vs-salt)
* [chef](https://www.redhat.com/en/topics/automation/ansible-vs-terraform)
* [AWS CloudFormation](https://www.redhat.com/en/topics/automation/ansible-vs-terraform)
* [ARM](https://www.redhat.com/en/topics/automation/ansible-vs-terraform)
* [Google Cloud Deployment Manager](https://www.redhat.com/en/topics/automation/ansible-vs-terraform)

### **Difference of Infrastructure as Code tools 🛠**

Why do we have so many different tools, can't we just use one IaC tool?

Well, no. Because no tool can do everything and each one is great in a specific area. IaC tools automate tasks in different categories for different phases:

#### **3 main task categories:**

* infrastructure provisioning
* configuration of provisioned infrastructure
* deployment of application

In infrastructure management, three key task categories can be identified:

### **1. Infrastructure Provisioning**

* **Definition**: This is the process of setting up the basic infrastructure needed for an application or system, such as virtual machines, storage, networks, and load balancers.
* **Tools**: Terraform, AWS CloudFormation, Azure Resource Manager (ARM).
* **Example**: Creating an EC2 instance on AWS or a VM on Azure.

### **2. Configuration of Provisioned Infrastructure**

* **Definition**: After provisioning, this involves configuring the systems, such as installing necessary software, setting up services, applying security settings, and configuring network rules.
* **Tools**: Ansible, Chef, Puppet, SaltStack.
* **Example**: Installing NGINX on a virtual machine and configuring firewall rules.

### **3. Deployment of Applications**

* **Definition**: Deploying the actual application code, setting up databases, and ensuring the application runs on the provisioned and configured infrastructure.
* **Tools**: Jenkins, Docker, Kubernetes, CI/CD pipelines.
* **Example**: Deploying a web application and configuring a database on provisioned servers.

#### **Distinction of phases:**

* initial setup phase
* maintaining phase

In infrastructure management, three key task categories can be identified:

### **1. Infrastructure Provisioning**

* **Definition**: This is the process of setting up the basic infrastructure needed for an application or system, such as virtual machines, storage, networks, and load balancers.
* **Tools**: Terraform, AWS CloudFormation, Azure Resource Manager (ARM).
* **Example**: Creating an EC2 instance on AWS or a VM on Azure.

### **2. Configuration of Provisioned Infrastructure**

* **Definition**: After provisioning, this involves configuring the systems, such as installing necessary software, setting up services, applying security settings, and configuring network rules.
* **Tools**: Ansible, Chef, Puppet, SaltStack.
* **Example**: Installing NGINX on a virtual machine and configuring firewall rules.

### **3. Deployment of Applications**

* **Definition**: Deploying the actual application code, setting up databases, and ensuring the application runs on the provisioned and configured infrastructure.
* **Tools**: Jenkins, Docker, Kubernetes, CI/CD pipelines.
* **Example**: Deploying a web application and configuring a database on provisioned servers.

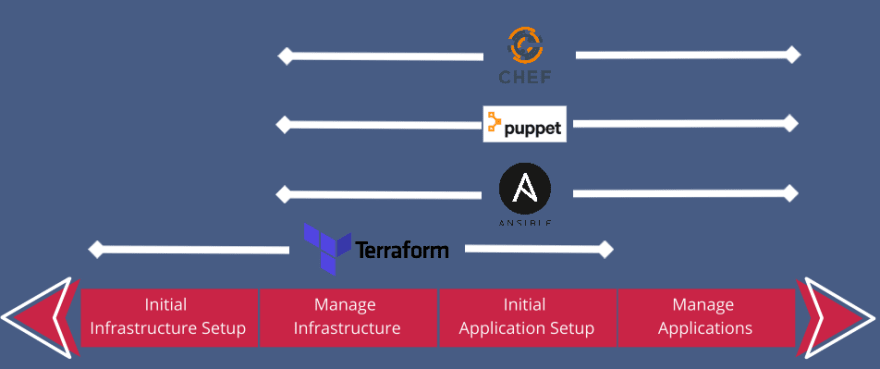
## **Distinction of Phases**

### **1. Initial Setup Phase**

* **Description**: This is the phase where the infrastructure and environment are set up for the first time.
* **Tasks**:
  + Provision infrastructure.
  + Configure infrastructure.
  + Initial installation of software .
  + Initial configuration of software .

### **2. Maintenance Phase**

* **Description**: Once the environment is live, this phase involves ongoing tasks to ensure stability, scalability, and updates.
* **Tasks**:
  + Scaling or updating infrastructure as needed.
  + updating software on provisioned infrastructure.
  + Add or remove servers.
  + Deploying new versions or updates of the application.



So, in most cases you would use a combination of 2 or more IaC tools.

* **Terraform to provision and configure infrastructure**
* **Ansible to install and deploy applications**

#### **Distinction in how they work**

Moreover, the IaC tools differ in the way HOW they work:

* declarative vs procedural
* mutable vs immutable
* agent vs agentless

Choosing depending on preferences and needs.That’s why we have multiple IAC tools.

**Key benefits of IaC include:**

