Terraform

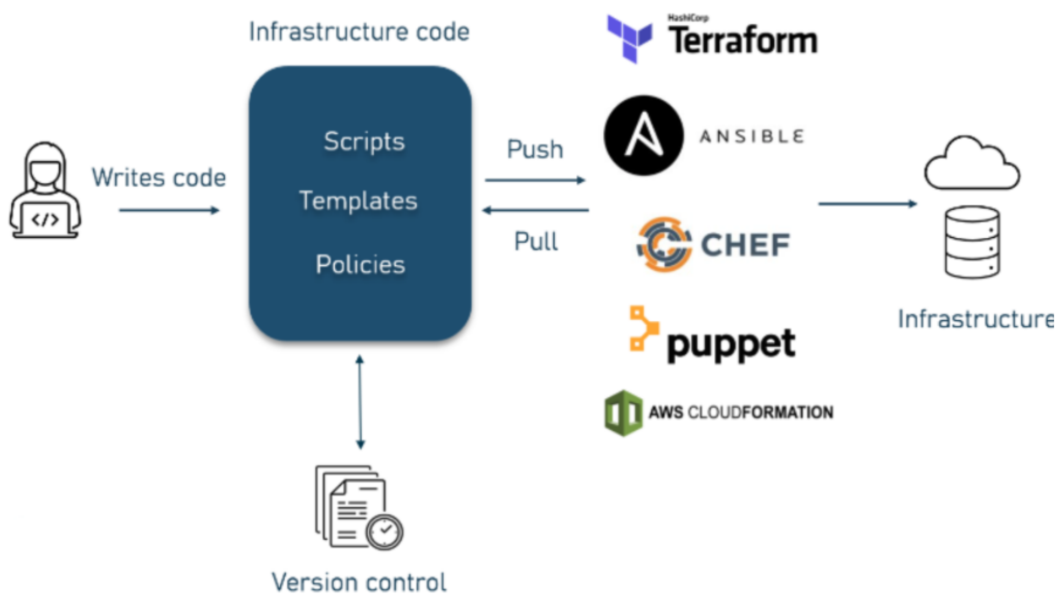
Traditional Infrastructure Deployment

Traditional infrastructure deployment involves manually configuring and provisioning physical servers, networking, and hardware components, often leading to longer setup times and increased room for human error. Main disadvantages include :

- Scalability Challenges
- Maintenance Complexity
- Inconsistencies
  -> two sperate teams: one for provisioning and one for deployment

Iac

Infrastructure as Code (IaC) is an approach that uses code to automate the provisioning and management of IT infrastructure. It allows for the declarative definition of infrastructure components, enabling rapid, consistent, and scalable deployments while promoting version control and reproducibility.



benefits

- INVOLVES LESS HUMAN ERRORS
- MORE AUTOMATION
- COLLABORATION BETWEEN OPS AND DEV
- INCREASES TRANSPARENCY
- TRACEABILITY
- INTEGRITY
- REPEATABILITY
- YOUR CODE IS GREAT DOCUMENTATION
- AGILITY

Terraform

- automate and manage infra , platform and services running on that platform
  - open source

    - declarative
- tool for infra provisioning

Terraform VS Ansible

Ansible		Terraform	
- mainly config tool for deploying apps once the infra is there - more mature		- mainly infra provisioning tool - relatively new -	

How does Terraform work

Architecture

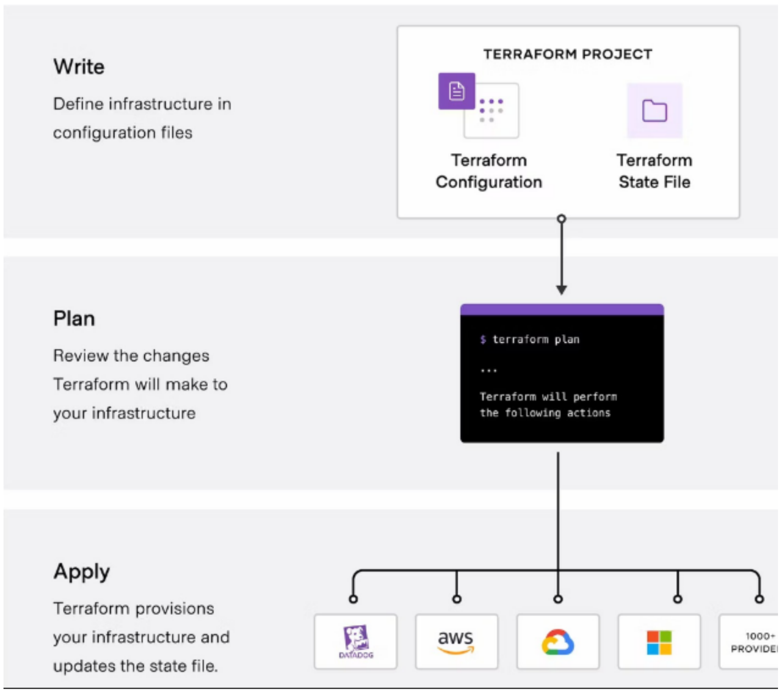
core component

TF has a core that takes as inout the current state and config files (the desired end result). It then compares the current state to the desired state and figures out what nedds to be done (created, updated / detroyed..) to get to the desired state

Providers

Iaas (the trength of terraform) , Paas, SaaS

Workflow


Write
Define infrastructure in configuration files
TERRAFORM PROJECT
Terraform Configuration
Terraform State File
Plan
Review the changes Terraform will make to your infrastructure
Apply
Terraform provisions your infrastructure and updates the state file.
DATADOG
aws
1000+ PROVIDE

```
$ terraform plan
...
Terraform will perform
the following actions
```

Commands

- `refresh` : get current state
- `plan` : create an execution plan (add server, add network ..) to get to the desired state
- `apply` : execute the plan
- `destroy` : revert what has been created