Terraform

Intro

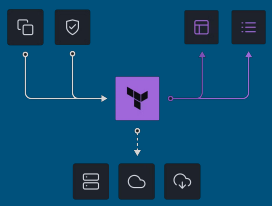
emerson@paduan.pro.br

O que é ?

Terraform é uma ferramenta de infraestrutura como código que permite construir, alterar e versionar infraestrutura com segurança e eficiência.

É uma ferramenta de código aberto, independente de provedor de nuvem, escrita na linguagem Go e criada pela HashiCorp.

O Terraform permite descrever sua infraestrutura completa na forma de código.



emerson@paduan.pro.br

Observação

Note: New versions of Terraform will be placed under the BUSL license, but everything created before version 1.5.x stays open-source. OpenTofu is an open-source version of Terraform that will expand on Terraform's existing concepts and offerings. It is a viable alternative to HashiCorp's Terraform, being forked from Terraform version 1.5.6. OpenTofu retained all the features and functionalities that had made Terraform popular among developers while also introducing improvements and enhancements.

Para quê ?

Uma das principais funções do Terraform é o provisionamento de nuvem pública em um dos principais provedores.

O segundo uso principal do Terraform é facilitar implantações em várias nuvens. Os engenheiros podem utilizar a mesma sintaxe sem se familiarizarem com diversas ferramentas e tecnologias.

O terceiro uso mais comum do Terraform é implantar, gerenciar e orquestrar recursos com provedores de nuvem personalizados.

Vantagens

Rapidez e Simplicidade.

Colaboração em equipe.

Redução de erros.

Recuperação de desastres.

Enhanced Security.

Competidores

Ansible é uma ferramenta IaC projetada para automatizar a configuração e o gerenciamento do sistema. Ansible não é uma ferramenta de provisionamento e segue uma abordagem processual, o que significa que o usuário precisa especificar manualmente as etapas de provisionamento. O Terraform permite o gerenciamento completo do ciclo de vida, enquanto o Ansible não.

Pulumi é uma ferramenta IaC de código aberto popular que pode ser usada para projetar, implantar e gerenciar recursos de infraestrutura em nuvem. Ao contrário do Terraform, Pulumi não usa uma linguagem de software específica de domínio, permitindo aos usuários implantar em GO, .NET, JavaScript e outros. Embora existam alguns benefícios de flexibilidade para o Pulumi, o Terraform é geralmente superior quanto maior o escopo de implantação.

Como funciona

O Terraform permite definir e gerenciar toda a sua infraestrutura por meio de arquivos de configuração e controle de versão.

Arquitetura

Terraform Core

Terraform Core usa duas fontes de entrada. A primeira é a entrada de origem que o usuário configura no Terraform, definindo quais recursos precisam ser criados ou provisionados. A segunda fonte de entrada consiste em dados alimentados no Terraform sobre a aparência da configuração atual da infraestrutura. O Terraform então pega essas informações e determina quais ações precisam ser tomadas.

O Terraform Core basicamente descobre o que precisa ser criado, atualizado ou excluído para provisionar totalmente sua infraestrutura.

Arquitetura

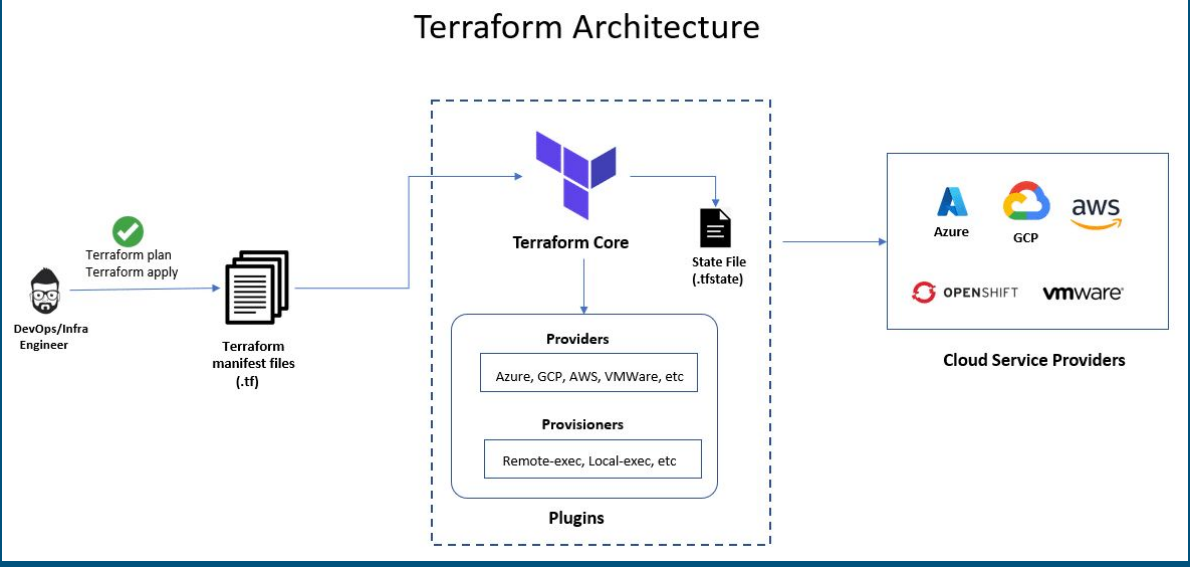
Terraform Providers

Normalmente são provedores de nuvem como AWS ou Azure, mas podem ser qualquer outra infraestrutura ou plataforma como ferramenta de serviço. Kubernetes, por exemplo, também é considerado um provedor utilizado pelo Terraform. Terraform possui mais de cem providers para diversas tecnologias.

Cada provider pode ser IaaS, PaaS ou SaaS, e dão acesso a mais de 1000 recursos, como AWS EC2, S3, etc.

Você pode então criar infraestrutura em diferentes níveis, empilhando Kubernetes sobre o Azure, por exemplo.

Arquitetura



Exemplo

```
# Configure the AWS Provider
provider "aws" {
  version = "~> 2.0"
  region  = "us-east-1"
}

# Create a VPC
resource "aws_vpc" "example" {
  cidr_block = "10.0.0.0/16"
}
```

O Terraform utiliza uma abordagem DECLARATIVA, e não imperativa, ou seja, dizemos O QUÊ desejamos, e não é necessário dizer COMO será feito para conseguir esse resultado.

Principais Comandos

*Refresh (deprecated *) - obtêm o estado atual da infraestrutura no provedor*

init - inicializa seu diretório Terraform e baixa os provedores para sua configuração.

plan - cria um plano de execução (o que precisa ser feito no provedor) para chegar o estado desejado

apply - executa o plano de execução. O Apply executa o refresh e o plan.

destroy - remove todos os recursos criados pelo Applay. Também executa o Refresh e o Plan.

Outros Comandos

fmt - ajusta a formatação do arquivo de configuração

validate - Verifica se a configuração é válida

show - Exibe os detalhes de estado atual

Terraform Registry

....

Instalação

emerson@paduan.pro.br

Variáveis

Criar um arquivo de variáveis

terraform apply -var "nome_variavel=valor"

Powershell env variables:
$env:AWS_ACCESS_KEY_ID='valor'
$env:AWS_SECRET_ACCESS_KEY='valor'

Prompt command:
set WS_ACCESS_KEY_ID=valor
set AWS_SECRET_ACCESS_KEY=valor

Bash:
export WS_ACCESS_KEY_ID=valor

emerson@paduan.pro.br