HashiCorp

Terraform

Terraform:
Automatizando seu caminho na nuvem!

Autor

Bryan Albuquerque, também conhecido como “Red Hat”.

Padawan de infraestrutura na **4Linux**, desenvolvedor nas horas vagas e apaixonado por open source desde sempre!

LinkedIn

GitHub

4LINUX
OPEN SOFTWARE SPECIALISTS

Agenda

1. Conceitos

- Infra moderna
- Infra como código
- Imutabilidade

2. Terraform

- O que é
- Como funciona
- Porque usar

3. Demo

- Overview
- Infra imutável

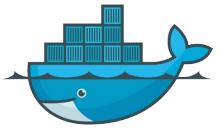
4. Dúvidas

Infraestrutura nos dias de hoje

- Cloud (Cloud native)
- Automação
- Orquestradores
- Containers
- Abstrações, muitas abstrações!
- **Linux <3 !**

Google Cloud Platform

kubernetes



docker

CLOUD NATIVE
COMPUTING FOUNDATION

Infra como código

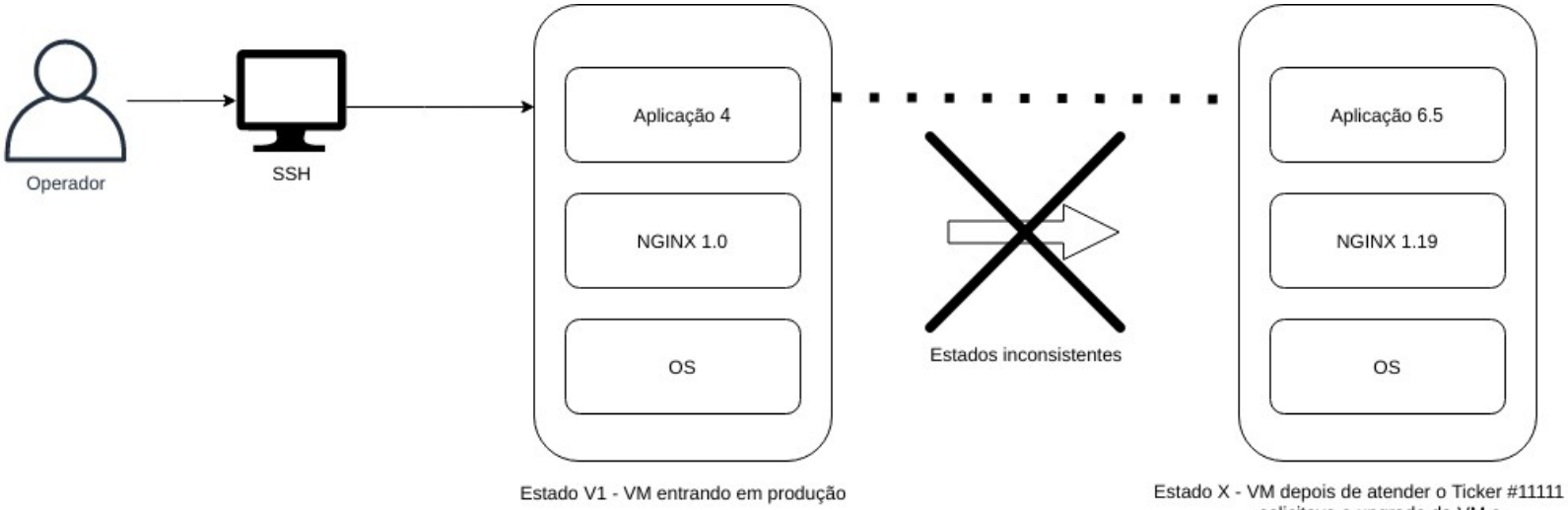
Em resumo, pense na sua infraestrutura como software

- Versionamento (Git)
- Testes
- Automação primeiro (Automation first)
- Reprodutibilidade e idempotência
- Conformidade (Compliance)
- Auditoria e histórico
- Sem mais painéis de Cloud diferentes e scripts _final.sh !

Infra imutável

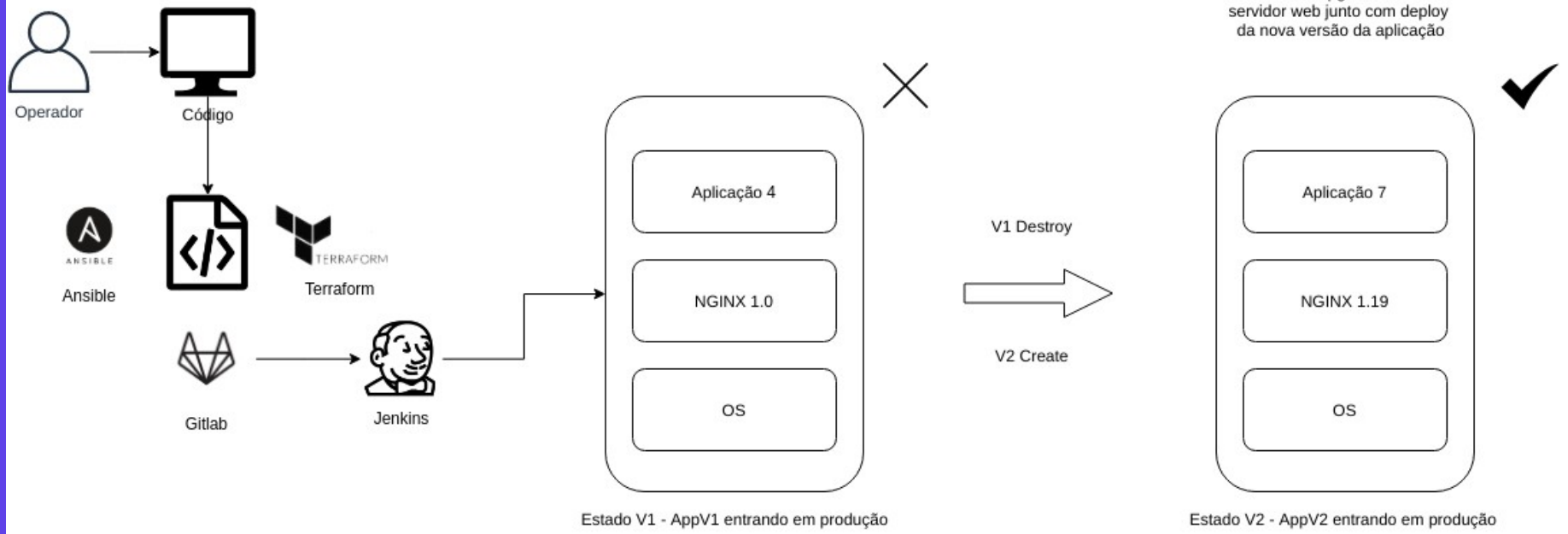
- Sem mais infra “pet”, nada de dar nome ao servidor de produção!
- Sem mais atualizações uma vez que a maquina esteja ativa (pelo menos não do jeito que você esta pensando)
- A infra inteira e versionada, e só ha entregas (releases) com versões impostas
- Consistente, confiável e previsível mas requer um time maduro
- Dados “stateful” ficam externalizados

Infra imutável



Estado V1 - VM entrando em produção

Estado X - VM depois de atender o Ticker #11111 que solicitava o upgrade da VM e servidor web junto com deploy da nova versão da aplicação



Estado V1 - AppV1 entrando em produção

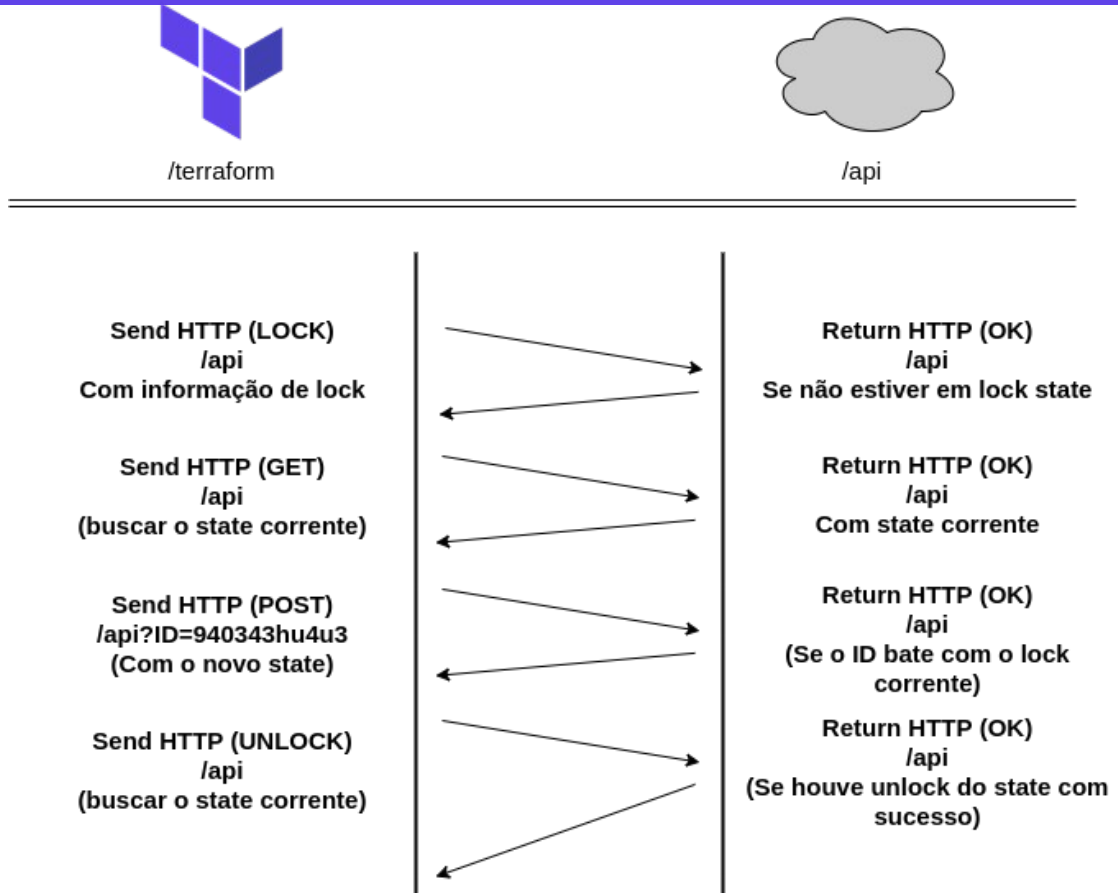
Estado V2 - AppV2 entrando em produção

Terraform – O que é

- Ferramenta para criar, alterar e criar versões de infraestrutura com segurança e eficiência baseadas no conceito de infraestrutura como código. Gerencia e orquestração de recursos em Cloud.
- Filosofia similar a do UNIX (O Tao da HashiCorp)
- Open Source
- Escrito em Golang
- Utiliza uma DSL (Domain Specific Language) declarativa muito próximo a linguagem humana, a HCL.
- Arquitetura baseada em plugins

Terraform – Como funciona

- Abstração em cima da API do provider
- Fluxo WPC
  - Write code
  - Plan changes
  - Create resources
  - Repeat



Terraform – Alternativas

Quanto a ferramentas que tem a mesma função do Terraform, nossa lista se resume basicamente a:

- Pulumi
- Gyro
- Cloudify
- Soluções do Cloud Provider (Cloud Formation, Deployment manager e etc)
- Soluções completamente baseada em código (Libcloud, boto e RESTFul APIs)

Terraform – Porque usar

- Sintaxe simples mais poderosa
- Melhor suporte de providers
- Melhor balanceamento entre abstração e flexibilidade
- O State do Terraform nos ajuda a manter um rastreio do código vs realidade
- Desenhada para a tarefa de provisionamento e orquestração
- Ganho de tempo exponencial em tarefas desempenhadas com a ferramenta
- “Democratiza” a infra

Demo - Overview

- Estrutura
- Recursos
- Variáveis
- Outputs
- Data sources
- Funções

HORA DO TERMINAL!

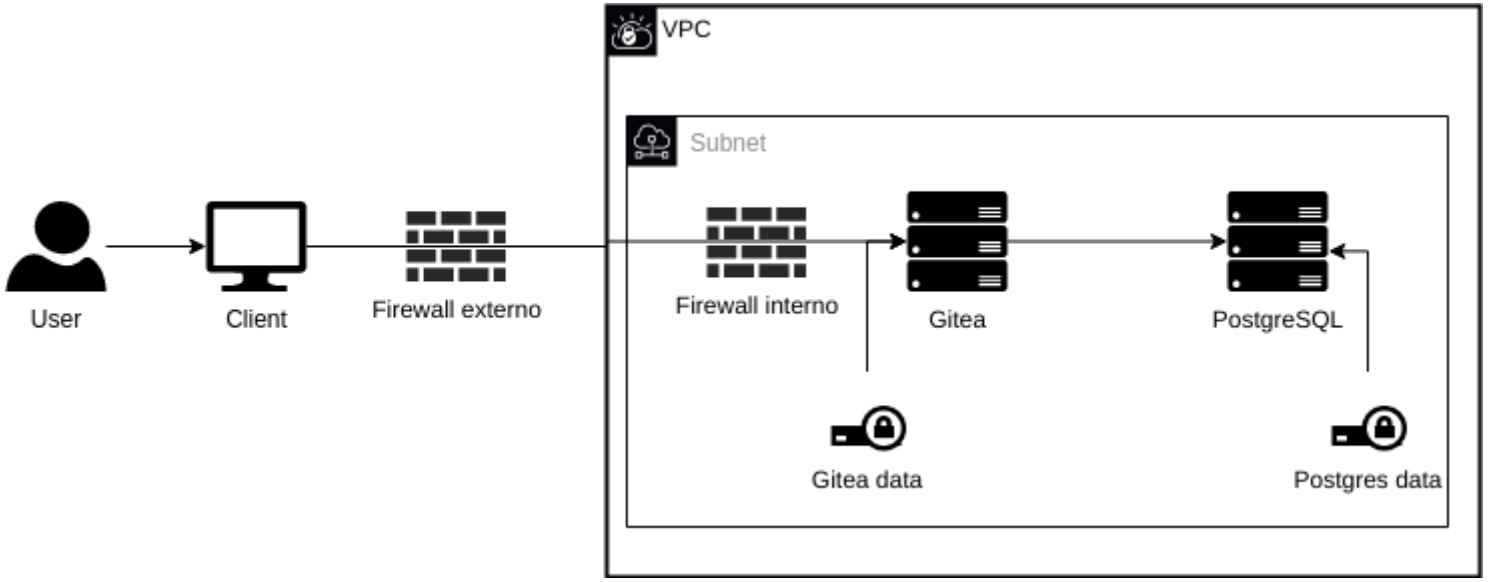
NixOS

Uma distribuição Linux com base no gerenciador de pacotes Nix

- Updates reproduzíveis e atômicos
- Rollbacks
- Configuração declarativa
- Source-based com um cache binário gigante
- Gerenciamento de pacotes multiusuário
- Stateless

Demo - Imutável

Arquitetura



HORA DO TERMINAL!

Dúvidas?

Fim!

Obrigado pela oportunidade!

LinkedIn

Curso de Terraform da 4Linux