unicast

/BOAS PRATICAS PARA PROJETOS DE TERRAFORM

RECDEV Summit 2023

MAIN.TF

unicast

<ANTONIO.JUNIOR>

Principal Cloud & DevOps Solution Architect
Microsoft MVP Azure

WHO.TF


unicast
/INFRAESTRUTURA
COMO CÓDGIO
COM TERRAFORM
TERRAFORM.TF

/O QUE É O TERRAFORM?

O Terraform é uma ferramenta de infraestrutura como código que permite criar, alterar e versionar a infraestrutura de maneira segura e eficiente.

- O Terraform é uma ferramenta open-source que permite criar e gerenciar a infraestrutura de maneira declarativa.
- Utiliza a linguagem HashiCorp Configuration Language (HCL) para descrever a infraestrutura desejada.
- É compatível com diversos provedores de nuvem e outros sistemas de infraestrutura.

unicast

/TERRAFORMANDO

/WRITE

Define a infraestrutura em arquivos de configuração

/PLAN

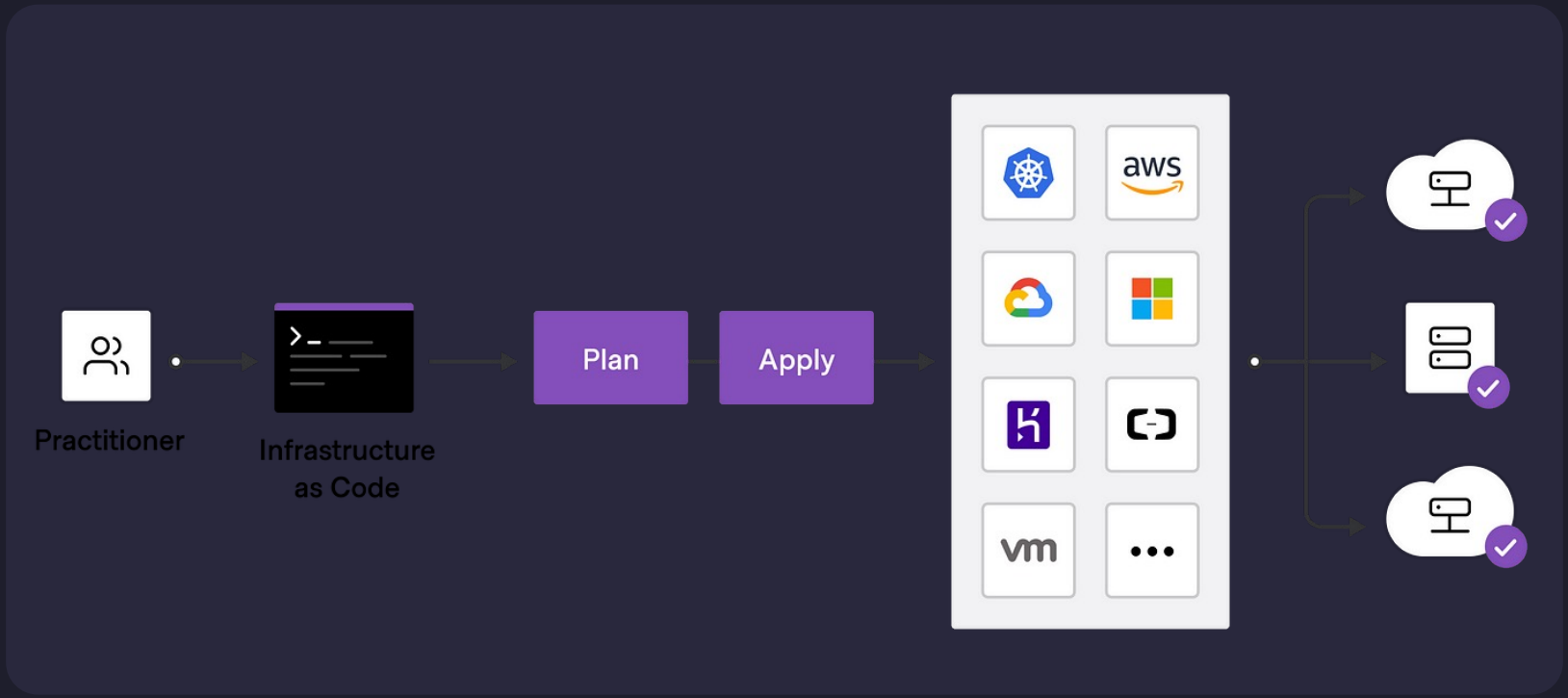
Revisa as mudanças que o Terraform irá aplicar em sua infraestrutura

/APPLY

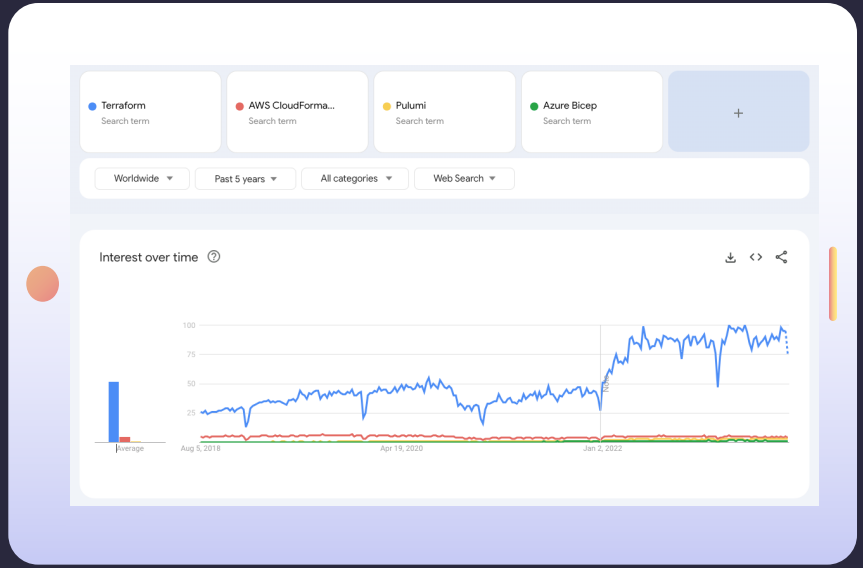
Terraform provisiona sua infraestrutura e atualiza arquivo de estado

unicast

/TERRAFORM WORKFLOW



/PORQUE TERRAFORM?




unicast
/BOAS PRÁTICAS
PARA PROJETOS
EM TERRAFORM
STAGE.TF

/REMOTE STATE

O **Remote State** é um recurso do Terraform que permite armazenar o estado da infraestrutura em um local centralizado, como um bucket do S3 na AWS ou um blob do Azure Storage na Microsoft Azure.

Isso pode ajudar a manter o estado da infraestrutura consistente em todas as instâncias do Terraform, o que pode ajudar a evitar erros e garantir que todos estejam trabalhando com a versão mais recente da configuração.

```
terraform {
 backend "azurerm" {
  resource_group_name = "tfstate-rg"
  storage_account_name = "tfstateacc"
  container_name = "tfstate"
  key = "terraform.tfstate"
 }
}
```

/PARTIAL CONFIGURATION

Você não precisa especificar todos os argumentos necessários na configuração de backend.

Omitir certos argumentos pode ser desejável se alguns argumentos forem fornecidos automaticamente por um script de automação executando o Terraform.

Quando alguns ou todos os argumentos são omitidos, chamamos isso de **partial configuration.**

```
terraform init \
-backend-config="resource_group_name" \
-backend-config="storage_account_name" \
-backend-config="container_name" \
-backend-config="key" \
-backend-config="subscription_id" \
-backend-config="access_key"
```

/NAMING CONVENTION

Usar uma **convenção de nomenclatura** consistente pode parecer uma prática simples, mas pode ajudar muito na organização e na legibilidade do seu código. Além disso, seguir uma convenção padrão pode ajudar a manter uma estrutura consistente em todos os projetos, facilitando a colaboração entre a equipe.

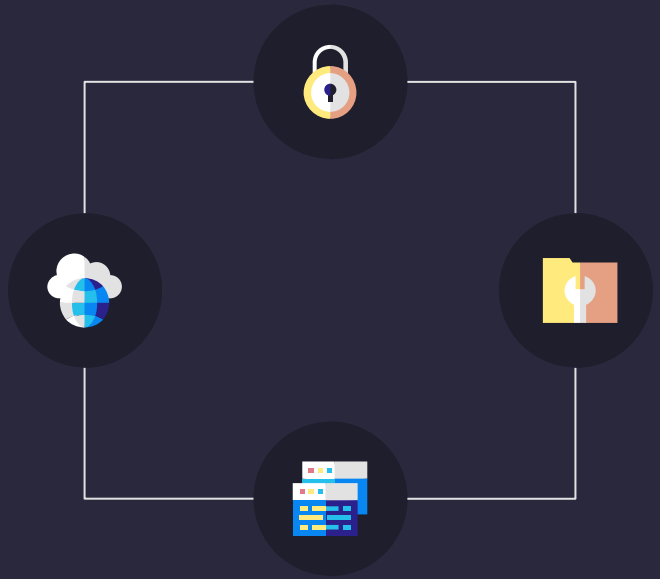
- Use letras minúsculas separadas por traços para nomes de recursos
- Utilize um prefixo para identificar o ambiente em que o recurso está sendo criado (por exemplo, “**prod-**“ para produção, “**dev-**“ para desenvolvimento)
- Use sufixos para identificar o tipo de recurso (por exemplo, “**-lb**” para load balancers, “**-vm**” para máquinas virtuais)
- Evite usar caracteres especiais ou abreviações confusas
- Mantenha a nomenclatura consistente em todos os projetos.

/DON'T REPEAT YOURSELF

O princípio **"Don't Repeat Yourself" (DRY)** prega eliminar a duplicação de código.

O Terraform permite a criação de módulos reutilizáveis que podem ser compartilhados entre diferentes projetos e ambientes.

Dessa forma, evita-se a repetição de código e facilita-se a manutenção e atualização da infraestrutura.



/TERRAFORM MODULES

/MÓDULOS

É recomendado o uso de módulospara compartilhar configurações comuns entre os ambientes. Essa abordagem facilita a manutenção, atualização e colaboração no código do Terraform.

- Pense em termos de abstração e agrupe recursos relacionados em um módulo
- Defina uma interface clara para o seu módulo, documentando o que o módulo faz e como ele deve ser usado
- Siga uma convenção de nomenclatura consistente para seus módulos
- Evite hard codes de variáveis em seus módulos
- Certifique-se de que seus módulos sejam testados e validados antes de serem usados em produção.

/DYNAMIC BLOCKS

Os **Dynamic Blocks** são uma das funcionalidades mais poderosas do Terraform. Eles permitem que você crie blocos de recursos dinâmicos com base em dados externos, o que pode ajudar a simplificar sua configuração e torná-la mais flexível.

- Pense em quais recursos podem se beneficiar do uso de Dynamic Blocks
- Use Dynamic Blocks para criar recursos que variam em tamanho ou dependem de dados externos
- Evite duplicação de código usando Dynamic Blocks em vez de criar recursos separados para cada instância ou elemento de dados
- Use a sintaxe correta para Dynamic Blocks, seguindo a documentação oficial do Terraform
- Certifique-se de que seus Dynamic Blocks estejam formatados e validados corretamente antes de implantar sua configuração.

/AKS_MODULE

```
resource "azurerm_kubernetes_cluster" "aks_cluster" {
# ...

dynamic "addon_profile" {
  for_each = var.enable_defender ? [1] : []
  content {
    oms_agent {
      enabled = true
}


enable_private_cluster = true
```

/LOOPS E CONDICIONAIS

/COUNT

O count é uma propriedade que pode ser definida em um bloco de recurso ou módulo, e que determina quantas instâncias desse recurso ou módulo serão criadas. Ele é útil quando você precisa criar várias instâncias do mesmo recurso ou módulo com configurações semelhantes.

/FOR_EACH

O for_each é um recurso mais avançado que permite criar várias instâncias de um recurso ou módulo com configurações diferentes. Ele é útil quando você precisa criar várias instâncias do mesmo recurso ou módulo com configurações diferentes.

/COUNT

```
resource "azurerm_virtual_machine" "vm" {
  count = 3

  name = "my-vm-${count.index}"
  location = var.location
  resource_group_name = var.resource_group_name
  network_interface_ids = [azurerm_network_interface.nic.id]

...
}
```

/FOR_EACH

```
locals {
  resources = {
   "resource1" = {
     name = "resource1"
     size = "small"
    },
   "resource2" = {
     name = "resource2"
     size = "large"
    },
   "resource3" = {
     name = "resource3"
     size = "medium"
    }
  }
}
```

```
resource "azurerm_virtual_machine" "vm" {
   for_each = local.resources

   name = "my-${each.value.name}-vm"
   vm_size = each.value.size
...
}
```

unicast

/TERRAFORM WORKSPACES

A estratégia de workspaces utiliza os recursos de workspaces do Terraform para gerenciar diferentes ambientes.

Cada workspace contém seu próprio estado e configurações, permitindo a alteração independente de cada ambiente.

Essa abordagem é útil quando os ambientes compartilham grande parte das configurações, com variações mínimas.

Essa estratégia envolve o uso de **workspaces** no **Terraform** para manter o código para cada ambiente em um único diretório.

Um **workspace** é uma instância isolada de um conjunto de recursos no **Terraform**. Cada **workspace** tem seu próprio estado.

Por padrão, o Terraform tem um único workspace chamado “**default**”, mas você pode criar novos workspaces para gerenciar recursos em diferentes ambientes.

/FERRAMENTAS ADICIONAIS

Existem diversas ferramentas adicionais que podem ajudar a aumentar o potencial do Terraform e melhorar ainda mais a experiência de desenvolvimento.

- **tflint**: é uma ferramenta de análise de código estático para Terraform que ajuda a identificar possíveis erros de sintaxe, estilo e configuração, além de fornecer sugestões para melhorar o código.
- **checkov**: é outra ferramenta de análise de código estático que ajuda a identificar riscos de segurança e conformidade em suas configurações de infraestrutura.
- **terratest**: é uma biblioteca de testes para infraestrutura em nuvem que permite criar testes automatizados para sua infraestrutura e validar se ela está funcionando conforme o esperado.
- **terraform-docs**: é uma ferramenta que ajuda a gerar documentação para seus módulos Terraform automaticamente, seguindo padrões de documentação estabelecidos.
- **infracost**: é uma plataforma de gerenciamento de custos para infraestrutura em nuvem que ajuda a monitorar e otimizar seus gastos com recursos na nuvem.

unicast

/OBRIGADO!

asilva@unicast.com.br
github.com/asilvajunior
unicast.com.br

Avalie esta apresentação, acesse o repositório e baixe o conteúdo apresentado!

THANKS.TF