



Lab: Terraform Estado Remoto

Para gerenciar seus recursos de infraestrutura de maneira adequada e correta, o Terraform armazena o estado de sua infraestrutura gerenciada. Cada configuração do Terraform pode especificar um back-end que define exatamente onde e como as operações são executadas. A maioria dos back-ends oferece suporte a recursos de segurança e colaboração, portanto, o uso de um back-end é obrigatório, tanto do ponto de vista da segurança quanto do trabalho em equipe.

O Terraform tem uma seleção integrada de back-ends e o back-end configurado deve estar disponível na versão do Terraform que você está usando. Neste laboratório, exploraremos o uso do backend remote utilizando S3.

- Tarefa 1: Backend Padrão S3

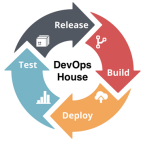
Backends Padrões

Os back-ends padrão do Terraform armazenam o estado remotamente e executam operações do terraform localmente por meio da interface de linha de comando. Os backends populares incluem:

- AWS S3 Backend (com DynamoDB)
- Google Cloud Storage Backend
- Azure Storage Backend

Consulte a documentação do Terraform para obter uma lista completa de backends padrão do Terraform.

A maioria dos back-ends também oferece suporte a recursos de colaboração, portanto, o uso de um back-end é obrigatório, tanto do ponto de vista da segurança quanto do trabalho em equipe. Nem todos esses recursos precisam ser configurados e ativados, mas mostraremos alguns dos itens mais úteis, incluindo controle de versão, criptografia.



Tarefa 1: Backend Padrão: S3

Passo 1.1 - Criar Bucket S3 e validar a configuração do Terraform

Acesse o S3 acessando o console AWS e crie um bucket para armazenar o estado do Terraform, ajuste o nome do seu bucket e deixe as demais propriedades como padrão. Se você estiver usando o S3 como back-end, convém configurar uma política IAM que conceda apenas acesso ao bucket do S3 para um pequeno grupo de pessoas confiáveis ou talvez apenas ao servidor CI que você usa para implantar em seus ambientes .

[Amazon S3](#) > [Buckets](#) > [Create bucket](#)

Create bucket [Info](#)

Buckets are containers for data stored in S3. [Learn more](#)

General configuration

Bucket name

Bucket name must be unique within the global namespace and follow the bucket naming rules. [See rules for bucket naming](#)

AWS Region

US East (N. Virginia) us-east-1

Copy settings from existing bucket - *optional*
Only the bucket settings in the following configuration are copied.

Choose bucket

Object Ownership [Info](#)

Control ownership of objects written to this bucket from other AWS accounts and the use of access control lists (ACLs). Object ownership determines who can specify access to objects.

☒ **ACLs disabled (recommended)**
All objects in this bucket are owned by this account. Access to this bucket and its objects is specified using only policies.

☐ **ACLs enabled**
Objects in this bucket can be owned by other AWS accounts. Access to this bucket and its objects can be specified using ACLs.

Object Ownership

Bucket owner enforced



Passo 1.2 - Validade o Estado do Backend S3

Queremos validar se podemos autenticar em nosso back-end terraform e listar as informações no arquivo de estado. Vamos validar se nossa configuração está apontando para o bucket e o caminho corretos. Inclua um block backend dentro do bloco terraform no arquivo `providers.tf`. Caso esteja utilizando um profile diferente do `default`, inclua a key `profile` especificando o profile utilizado,

```
terraform {  
  backend "s3" {  
    bucket = "terraform-state-xjulio"  
    key    = "dev/terraform.tfstate"  
    region = "us-east-1"  
    profile = "admin-2023"  
  }  
}
```

Se já estivermos inicializado o Terraform com o backend local e não tivermos nenhum recurso criar, teremos que inicializar novamente com a flag `-reconfigure`

```
terraform init -reconfigure
```


Caso existam recursos criados no estado local, então teremos que executar o commando para inicializar utilizando a flag `-migrate-state`

```
terraform init -migrate-state
```



Passo 1.3 - Habilitar versionamento no bucket S3

A ativação do controle de versão em nosso back-end terraform é importante, pois nos permite restaurar a versão anterior do estado, caso seja necessário. O back-end s3 oferece suporte ao controle de versão, portanto, todas as revisões do seu arquivo de estado são armazenadas.



Amazon S3 > Buckets > terraform-state-xjudio > Edit Bucket Versioning

Edit Bucket Versioning [Info](#)

Bucket Versioning

Versioning is a means of keeping multiple variants of an object in the same bucket. You can use versioning to preserve, retrieve, and restore every version of every object stored in your Amazon S3 bucket. With versioning, you can easily recover from both unintended user actions and application failures. [Learn more](#)

Bucket Versioning

☐ Suspend
This suspends the creation of object versions for all operations but preserves any existing object versions.

☒ Enable

ⓘ After enabling Bucket Versioning, you might need to update your lifecycle rules to manage previous versions of objects.

Multi-factor authentication (MFA) delete

An additional layer of security that requires multi-factor authentication for changing Bucket Versioning settings and permanently deleting object versions. To modify MFA delete settings, use the AWS CLI, AWS SDK, or the Amazon S3 REST API. [Learn more](#)

Disabled

Cancel [Save changes](#)

Depois que o controle de versão estiver ativado em seu bucket, executar o commando `terraform apply`.



HashiCorp Terraform Hands-On Labs



Agora você pode ver que seu arquivo de estado foi atualizado e, se marcar Mostrar versões no bucket, verá as diferentes versões de seu arquivo de estado.

Amazon S3 > Buckets > terraform-state-xjudio > dev/

dev/ Copy S3 URI

Objects Properties

Objects (2)
Objects are the fundamental entities stored in Amazon S3. You can use [Amazon S3 inventory](#) to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. [Learn more](#)

Refresh Copy S3 URI Copy URL Download Open Delete Actions Create folder Upload

☒ Show versions < 1 > ⚙

<input type="checkbox"/>	Name	Type	Version ID	Size	Storage class
<input type="checkbox"/>	terraform.tfstate	tfstate	NliVHn3DT6 .odJJNrt7h mWCN1sdjH efd	16.0 KB	Standard
<input type="checkbox"/>	terraform.tfstate	tfstate	FREHjIDVLC _x5GS4OXn K76NBqx.fE 83.	15.7 KB	Standard

Passo 1.4 - Habilitar criptografia no Bucket S3

Também é extremamente importante proteger os dados do estado de terraform, pois podem conter informações extremamente confidenciais. Armazene o estado do Terraform em um back-end compatível com criptografia. Em vez de armazenar seu estado em um arquivo local terraform.tfstate. Muitos back-ends suportam criptografia, de modo que, em vez de seus arquivos de estado estarem em texto simples, eles sempre serão criptografados, tanto em trânsito (por exemplo, via TLS) quanto em disco (por exemplo, via AES-256). O back-end s3 oferece suporte à criptografia, o que reduz as preocupações com o armazenamento de dados confidenciais em arquivos de estado.

Amazon S3 > Buckets > terraform-state-xjudio > Edit default encryption

Edit default encryption [Info](#)

Default encryption
Server-side encryption is automatically applied to new objects stored in this bucket.

Encryption type [Info](#)

- ☒ Server-side encryption with Amazon S3 managed keys (SSE-S3)
- ☐ Server-side encryption with AWS Key Management Service keys (SSE-KMS)
- ☐ Dual-layer server-side encryption with AWS Key Management Service keys (DSSE-KMS)
Secure your objects with two separate layers of encryption. For details on pricing, see [DSSE-KMS pricing](#) on the [Storage](#) tab of the [Amazon S3 pricing page](#).

Bucket Key
Using an S3 Bucket Key for SSE-KMS reduces encryption costs by lowering calls to AWS KMS. S3 Bucket Keys aren't supported for DSSE-KMS. [Learn more](#)

- ☐ Disable
- ☒ Enable

Cancel Save changes



HashiCorp Terraform
Hands-On Labs



Qualquer pessoa em sua equipe que tenha acesso a esse bucket do S3 poderá ver os arquivos de estado em um formato não criptografado, portanto, essa ainda é uma solução parcial, mas pelo menos os dados serão criptografados em repouso (o S3 oferece suporte ao server-side criptografia usando AES-256) e em trânsito (Terraform usa SSL para ler e gravar dados no S3). Uma recomendação seria utilizar chaves KMS e atribuir políticas de acesso à chave de criptografia.