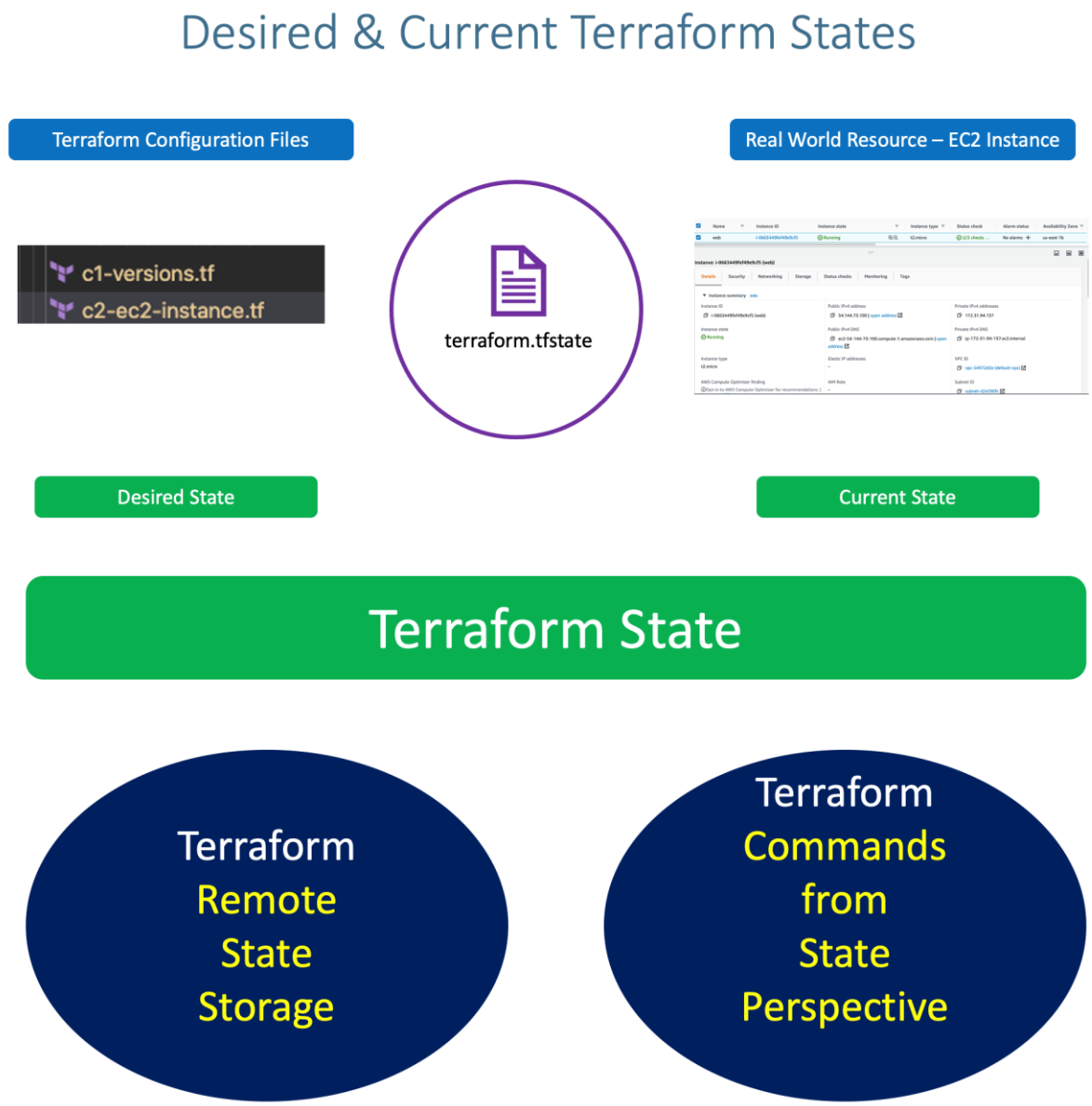
Terraform Remote State Storage and State Locking with AWS S3 and DynamoDB :

Step-01: Introduction

- Understand Terraform Backends
- Understand about Remote State Storage and its advantages
- This state is stored by default in a local file named "terraform.tfstate", but it can also be stored remotely, which works better in a team environment.
- Create AWS S3 bucket to store terraform.tfstate file and enable backend configurations in terraform settings block
- Understand about State Locking and its advantages
- Create DynamoDB Table and implement State Locking by enabling the same in Terraform backend configuration



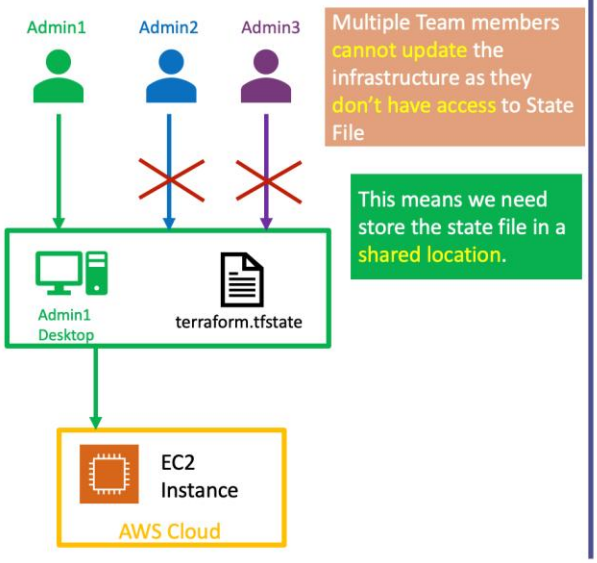
What is Terraform Backend ?

Backends are responsible for storing state and providing an API for state locking.

AWS S3 Bucket

AWS DynamoDB

Local State File



Multiple Team members cannot update the infrastructure as they don't have access to State File

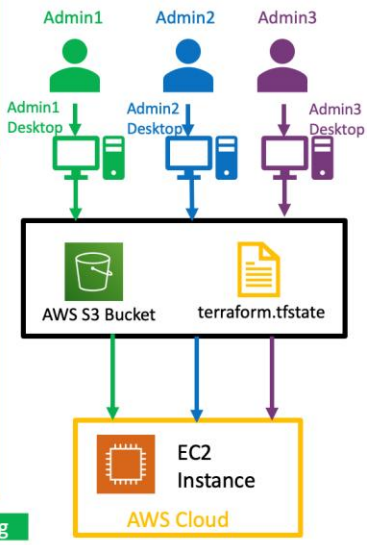
This means we need store the state file in a shared location.

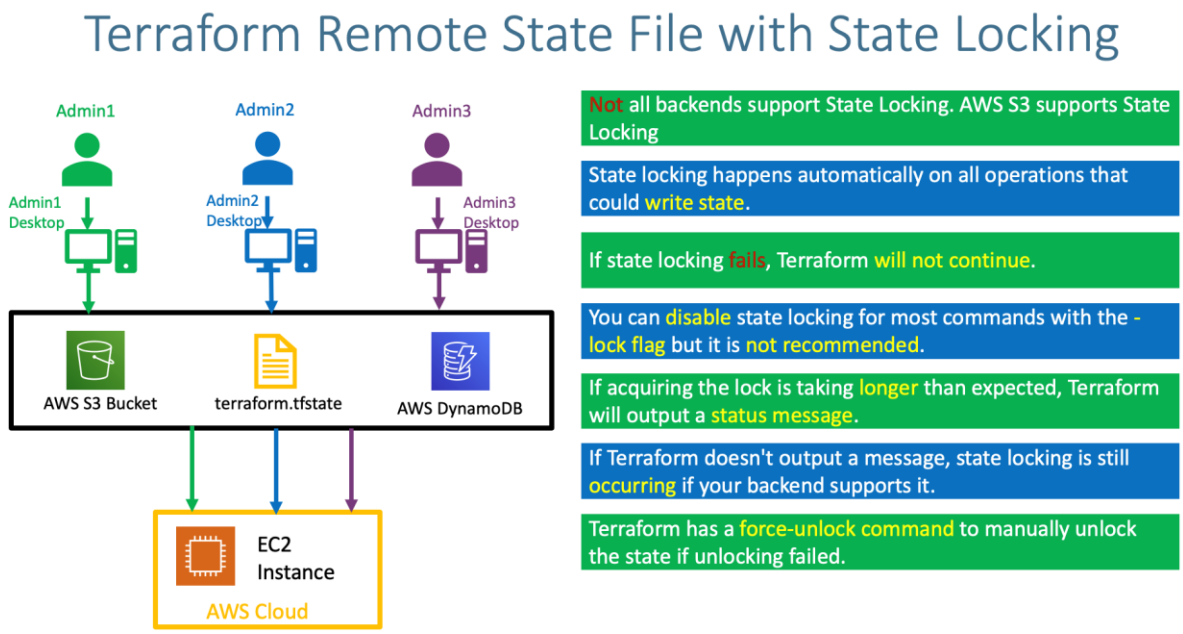
Remote State File

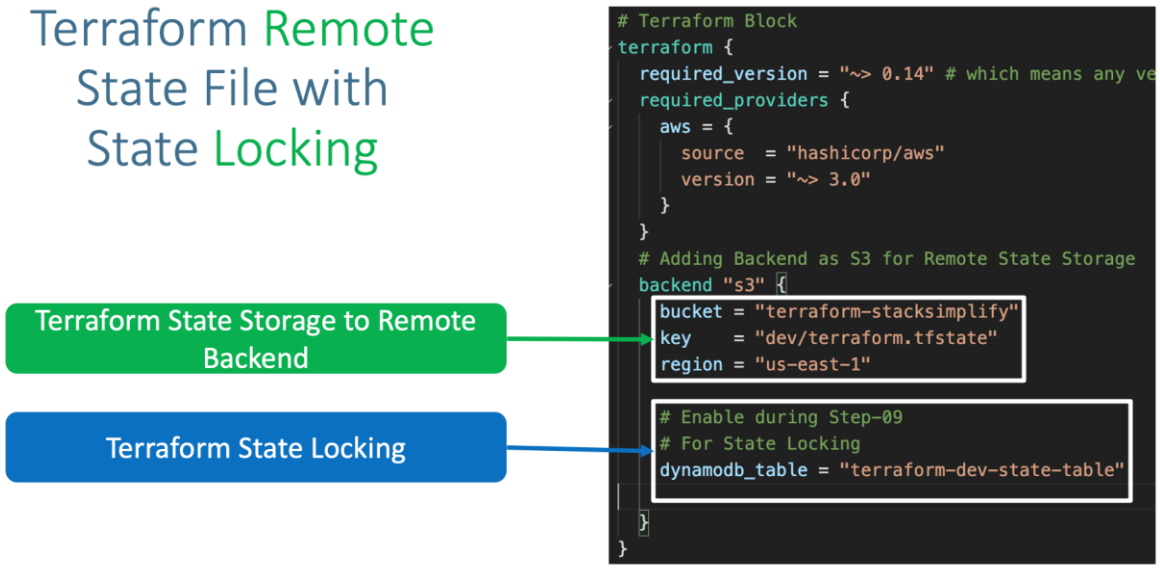
Using Terraform Backend concept we can use AWS S3 as the shared storage for State Files

If two team members are running Terraform at the same time, you may run into race conditions as multiple Terraform processes make concurrent updates to the state files, leading to conflicts, data loss, and state file corruption.

Implement State Locking







Step-02: Create S3 Bucket

- Go to Services -> S3 -> Create Bucket
- Bucket name: terraform-on-aws-for-ec2
- Region: US-East (N.Virginia)
- Bucket settings for Block Public Access: leave to defaults
- Bucket Versioning: Enable
- Rest all leave to defaults
- Click on Create Bucket
- Create Folder

    - Folder Name: dev
    - Click on Create Folder
- Create Folder
    - Folder Name: dev/project1-vpc
    - Click on Create Folder

Step-03: Terraform Backend Configuration

- Reference Sub-folder: terraform-manifests
- Terraform Backend as S3
- Add the below listed Terraform backend block in Terrafrom Settings block in main.tf

```
 # Adding Backend as S3 for Remote State Storage
 backend "s3" {
   bucket = "terraform-on-aws-for-ec2"
   key    = "dev/project1-vpc/terraform.tfstate"
   region = "us-east-1"

   # Enable during Step-09
   # For State Locking
   dynamodb_table = "dev-project1-vpc"
 }
```

Step-04: Terraform State Locking Introduction

- Understand about Terraform State Locking Advantages

Step-05: Add State Locking Feature using DynamoDB Table

- Create Dynamo DB Table
    - Table Name: dev-project1-vpc
    - Partition key (Primary Key): LockID (Type as String)
    - Table settings: Use default settings (checked)
    - Click on Create

Step-06: Execute Terraform Commands

```
# Initialize Terraform
terraform init
```

Observation:

Successfully configured the backend "s3"! Terraform will automatically

use this backend unless the backend configuration changes.

Terraform Validate

terraform validate

Review the terraform plan

terraform plan

Observation:

1) Below messages displayed at start and end of command

Acquiring state lock. This may take a few moments...

Releasing state lock. This may take a few moments...

2) Verify DynamoDB Table -> Items tab

Create Resources

terraform apply -auto-approve

Verify S3 Bucket for terraform.tfstate file

dev/project1-vpc/terraform.tfstate

Observation:

1. Finally at this point you should see the terraform.tfstate file in s3 bucket

2. As S3 bucket version is enabled, new versions of `terraform.tfstate` file new versions will be created and tracked if any changes happens to infrastructure using Terraform Configuration Files

Step-07: Destroy Resources

- Destroy Resources and Verify Bucket Versioning

Destroy Resources

terraform destroy -auto-approve

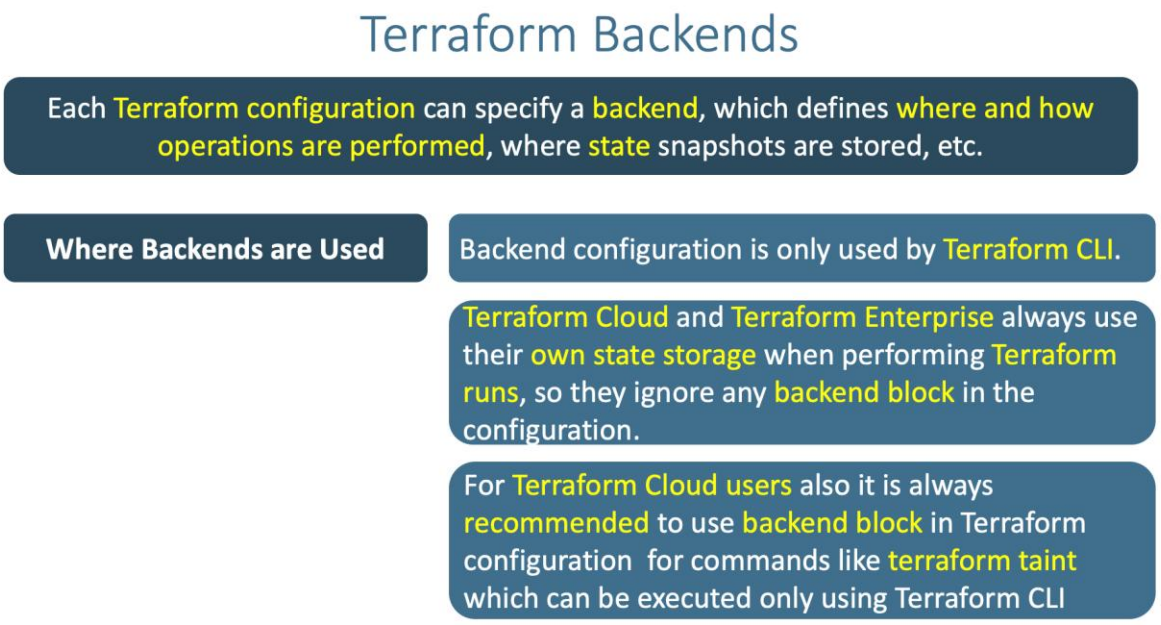
Clean-Up Files

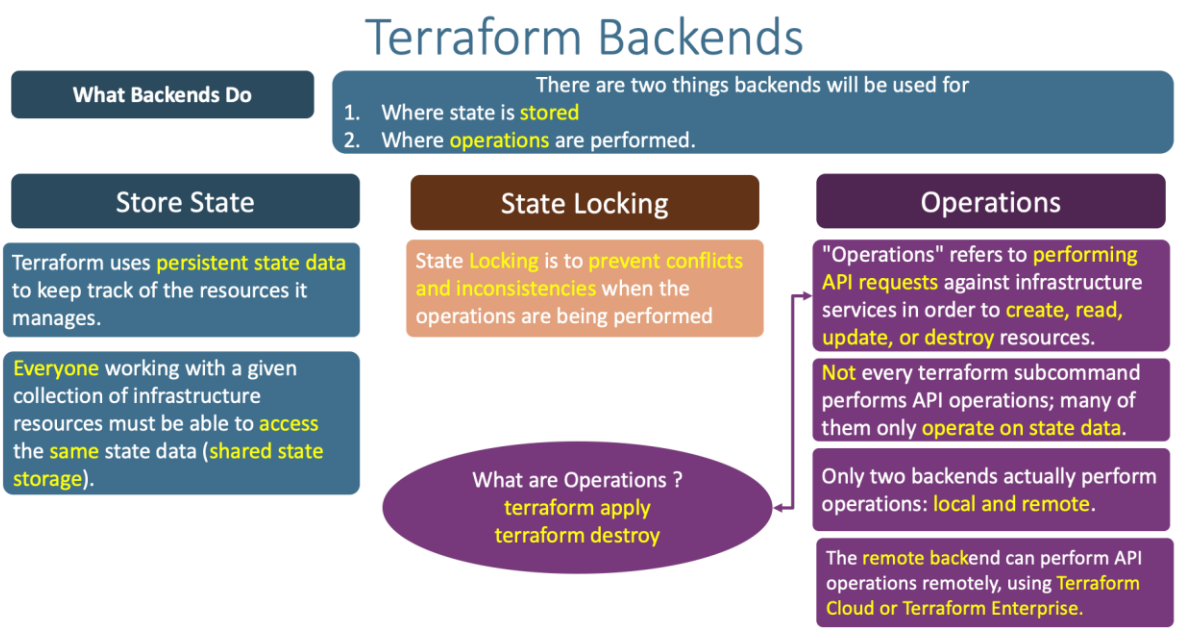
rm -rf .terraform*

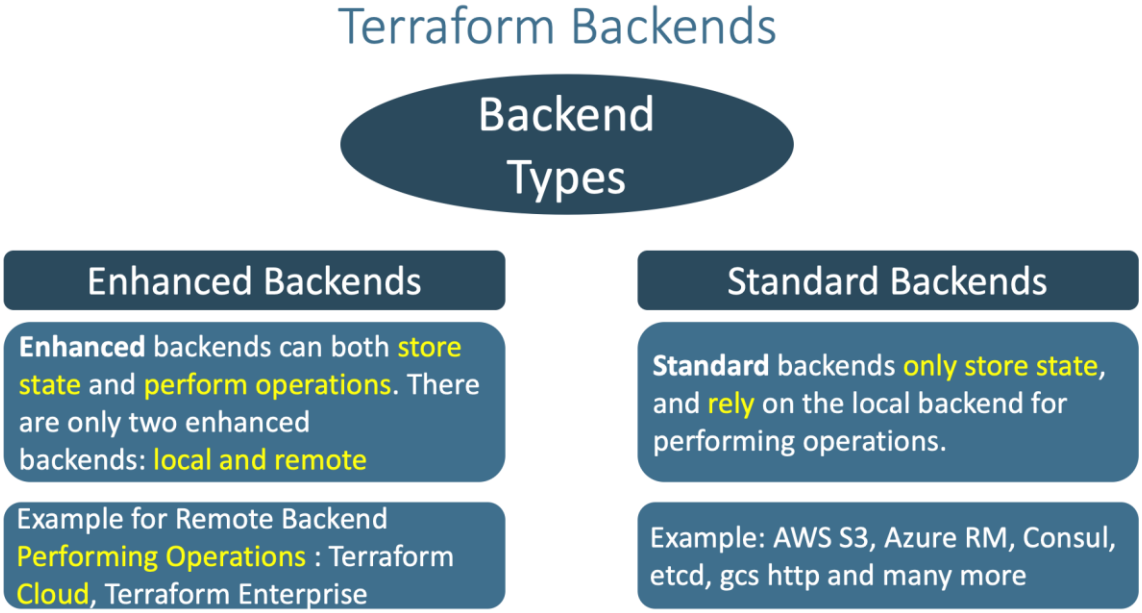
rm -rf terraform.tfstate* # This step not needed as e are using remote state storage here

Step-08: Little bit theory about Terraform Backends

- Understand little bit more about Terraform Backends
- Where and when Terraform Backends are used ?
- What Terraform backends do ?
- How many types of Terraform backends exists as on today ?







References

- AWS S3 Backend
- Terraform Backends
- Terraform State Storage
- Terraform State Locking
- Remote Backends - Enhanced