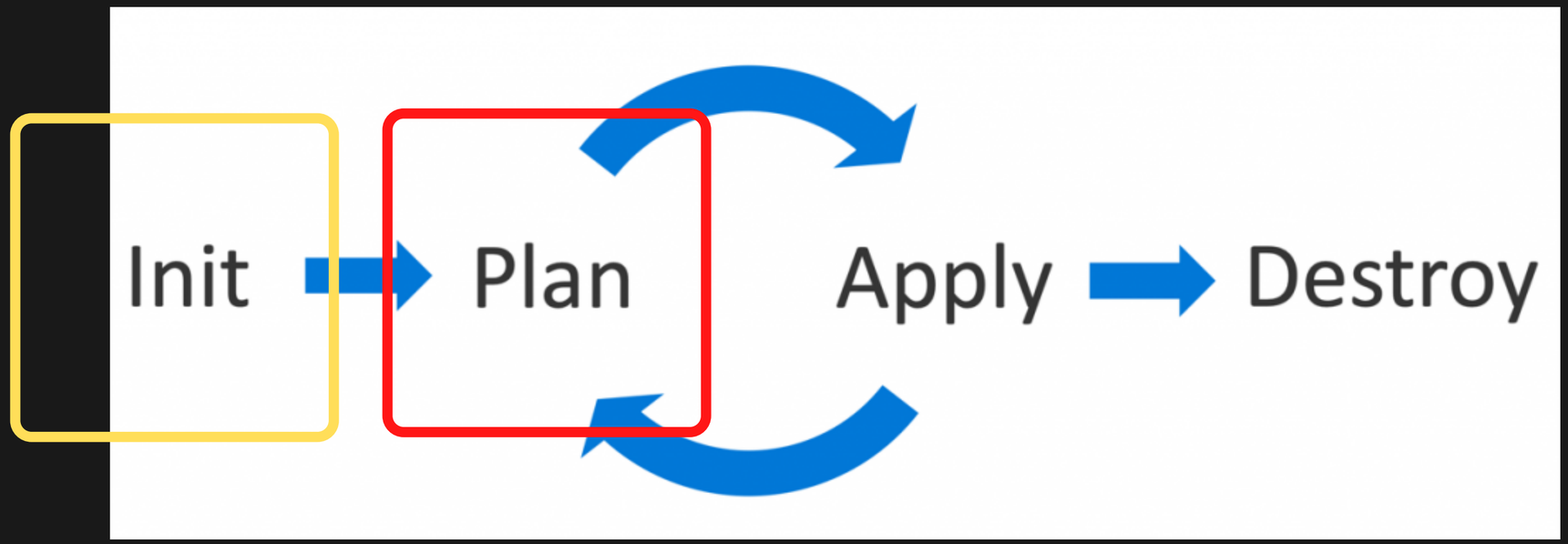
Terraform Plan

Terraform Plan



HashiCorp Terraform

Terraform Plan

- terraform plan - compares infrastructure defined in your code vs. the real world, but does not apply them
  - Dry run - does not apply the changes. Preview your changes before you make them
  - Safe to run multiple times
  - We call this, operational confidence
- Usually ran before terraform apply
- Always a good idea to run terraform plan before you permanently make any changes
- Terraform creates what is called an execution plan
  - A detailed output providing a report of what changes will take place
  - Output is rendered to stdout on your terminal
- There are three types of proposed changes:
  - In place updates
  - Complete updates
  - Creating Resources

Terraform Plan

```
  + resource "aws_security_group_rule" "ingress_allow_ssh" {
      + cidr_blocks              = [
          + "0.0.0.0/0",
        ]
      + description              = "SSH from home"
      + from_port                = 22
      + id                       = (known after apply)
      + ipv6_cidr_blocks         = [
          + "::/0",
        ]
      + protocol                 = "tcp"
      + security_group_id        = (known after apply)
      + self                     = false
      + source_security_group_id = (known after apply)
      + to_port                  = 22
      + type                     = "ingress"
    }

Plan: 3 to add, 0 to change, 0 to destroy.
```

HashiCorp Terraform

Terraform Plan

```
Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following symbols:
  + create

Terraform will perform the following actions:

  # aws_instance.main will be created
  + resource "aws_instance" "main" {
      + ami                          = "ami-013a129d325529d4d"
      + arn                          = (known after apply)
      + associate_public_ip_address  = false
      + availability_zone            = (known after apply)
      + cpu_core_count               = (known after apply)
      + cpu_threads_per_core         = (known after apply)
      + get_password_data            = false
      + host_id                      = (known after apply)
      + id                           = (known after apply)
      + instance_state               = (known after apply)
      + instance_type                = "t3.micro"
      + ipv6_address_count           = (known after apply)
      + ipv6_addresses               = (known after apply)
      + key_name                     = (known after apply)
      + outpost_arn                  = (known after apply)
      + password_data                = (known after apply)
      + placement_group              = (known after apply)
      + primary_network_interface_id = (known after apply)
      + private_dns                  = (known after apply)
      + private_ip                   = (known after apply)
      + public_dns                   = (known after apply)
      + public_ip                    = (known after apply)
      + secondary_private_ips        = (known after apply)
      + security_groups              = (known after apply)
      + source_dest_check            = true
      + subnet_id                    = (known after apply)
      + tags                         = {
          + "Name"        = "levelup_with_terraform-instance"
          + "course"      = "levelup_with_terraform"
          + "environment" = "dev"
```

HashiCorp Terraform

Terraform Plan

```
PS C:\Users\tekke\Desktop\levelup_with_terraform\workflow_basics> terraform plan
aws_security_group.allow_ssh: Refreshing state... [id=sg-089ca9f6c9c77f476]
aws_security_group_rule.ingress_allow_ssh: Refreshing state... [id=sgrule-3785398798]
aws_instance.main: Refreshing state... [id=i-06986e93352407fe9]

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following symbols:
-/+ destroy and then create replacement

Terraform will perform the following actions:

  # aws_instance.main must be replaced
-/+ resource "aws_instance" "main" {
      ~ arn                          = "arn:aws:ec2:us-west-2:320193497904:instance/i-06986e93352407fe9" -> (known after apply)
      ~ availability_zone            = "us-west-2b" -> (known after apply)
      ~ cpu_core_count               = 1 -> (known after apply)
      ~ cpu_threads_per_core         = 2 -> (known after apply)
      - disable_api_termination      = false -> null
      - ebs_optimized                = false -> null
      - hibernation                  = false -> null
      + host_id                      = (known after apply)
      ~ id                           = "i-06986e93352407fe9" -> (known after apply)
      ~ instance_state               = "running" -> (known after apply)
      ~ instance_type                = "t3.micro" -> "t3.small"
      ~ ipv6_address_count           = 0 -> (known after apply)
      ~ ipv6_addresses               = [] -> (known after apply)
      + key_name                     = (known after apply)
      - monitoring                   = false -> null
      + outpost_arn                  = (known after apply)
      + password_data                = (known after apply)
      + placement_group              = (known after apply)
      ~ primary_network_interface_id = "eni-01a90e5070645ec18" -> (known after apply)
      ~ private_dns                  = "ip-172-31-30-226.us-west-2.compute.internal" -> (known after apply)
      ~ private_ip                   = "172.31.30.226" -> (known after apply)
      ~ public_dns                   = "ec2-54-191-157-186.us-west-2.compute.amazonaws.com" -> (known after apply)
      ~ public_ip                    = "54.191.157.186" -> (known after apply)
      ~ secondary_private_ips        = [] -> (known after apply)
      ~ security_groups              = [
          - "allow_ssh",
        ] -> (known after apply)
      ~ subnet_id                    = "subnet-01935c521a639cf87" -> (known after apply)
        tags                         = {
```

HashiCorp Terraform

Terraform Plan

```
Terraform used the selected providers to generate the following execution plan.
 ~ update in-place

Terraform will perform the following actions:

 # aws_instance.main will be updated in-place
 ~ resource "aws_instance" "main" {
       id                           = "i-06986e93352407fe9"
     ~ tags                         = {
         + "Foo"         = "Bar"
           # (5 unchanged elements hidden)
       }
       # (27 unchanged attributes hidden)



       # (3 unchanged blocks hidden)
   }

Plan: 0 to add, 1 to change, 0 to destroy.
```

HashiCorp Terraform

Terraform Plan

- In place updates - performs updates without requiring to re-create the entire resource
  - When resource already exists, but change is found in code
  - Terraform applies your changes in-place
  - Reduces unintended side effects
  - Your resource remains functional with no downtime
- Complete update - update requires re-creating the entire resource
  - When resource already exists, but change is found in the code
  - Terraform is required to completely re-create the resource to apply changes
  - May lead to side effects, important to carefully review
  - May introduce downtime, as resource is destroyed and recreated
- Creating resources - Terraform detects a new resource
  - Brand new resource is detected, Terraform wants to create it

Terraform Plan

- How does Terraform know the type of update?
- Terraform relies on two critical components, to help make informed decisions:
  - Consulting the state file
  - Communicating with the provider
- Talk more in depth about state & providers, but overview:
  - State files - holds the current state of your Terraform environment/configuration. Contains metadata, attributes, and dependencies for all your defined resources. The brains of Terraform.
  - Providers - plugins which expose set of resources for Terraform to use. For example, the AWS Provider allows you to use and create AWS specific resources.
- Terraform consults your current state to view what resources are created, and what their attributes are
  - Helps understand what type of change it needs to make