

Terraform Workflows



Agenda

- Run Workflows
- Terraform Modules
- Workspaces
- Variables
- Git Repo Structure
- Q+A

The image features a dark blue background with decorative geometric patterns. In the top-left corner, there are several overlapping squares and rectangles filled with a fine grid of small dots, with some areas having diagonal lines. In the bottom-right corner, there is a large square filled with a fine grid of small dots.

Runs Workflows

Run Workflows

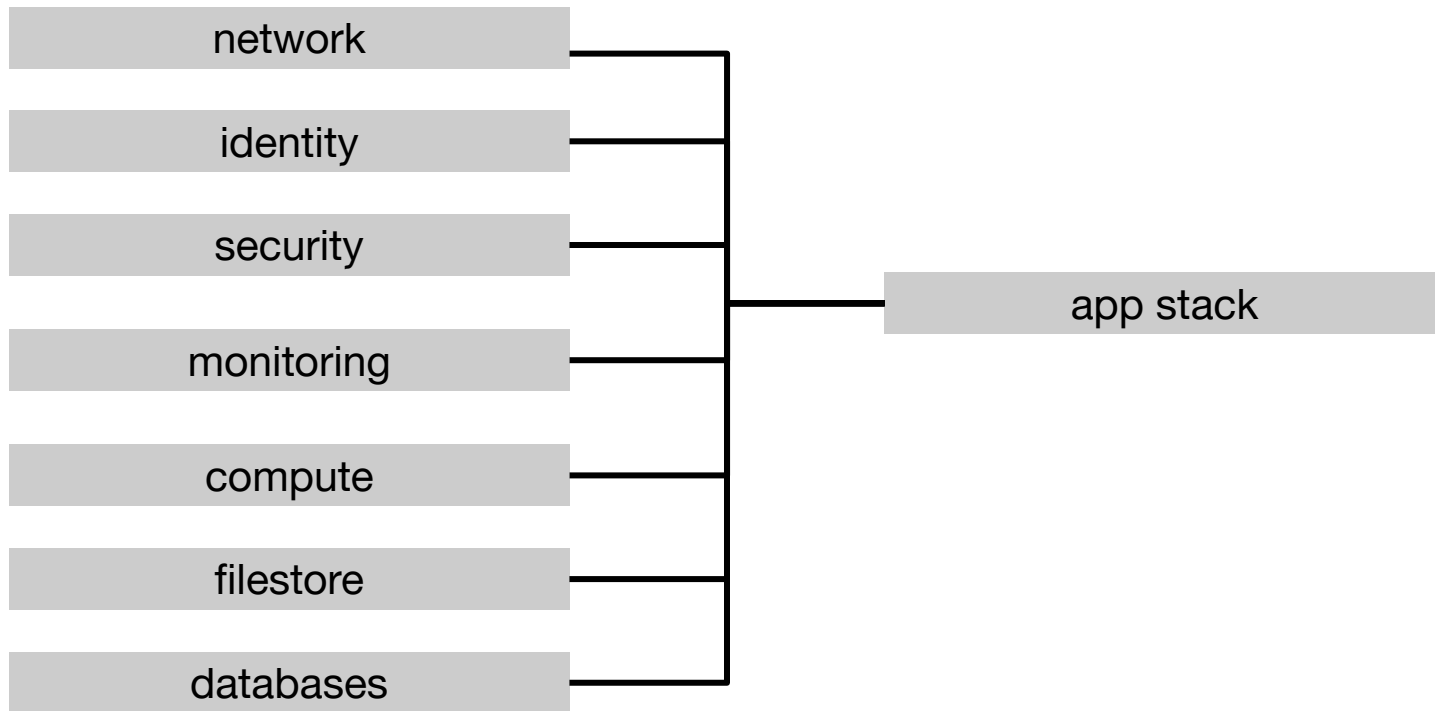


- [UI-Driven Runs](#) - manually trigger runs from the TFC web UI.
- [VCS-Driven Runs](#) - easiest integration, directly connects a Git Repo to a Terraform Workspace, with automatic runs on Git Commit and Pull Request code changes.
- [CLI-Driven Runs](#) - easy to use, single CLI command to trigger runs, takes files in the local folder, creates a .zip file, and sends the contents to the TFC API.
- [SDK-Driven Runs](#) - calls to the TFC API, using a Language Specific integration, available for Golang, Python, and .NET.
- [API-Driven Runs](#) - full control, all features available to the web UI have an API call, but requires custom coding JSON REST HTTP API calls.

The image features a dark blue background with decorative geometric patterns. In the top-left corner, there are several overlapping squares and rectangles filled with a fine grid of small white dots. In the bottom-right corner, there are similar shapes, but they are filled with a pattern of thin, parallel white lines. The text 'Terraform Modules' is centered in the middle of the image in a large, bold, white sans-serif font.

Terraform Modules

Architecture



Code Layout



Static Variables and Dynamically Generated Outputs can be passed between Modules.

```
# ./main.tf
variable "vpc_cidr" {
  default    = "10.0.0.0/16"
}

module "network" {
  source      = "./network"
  vpc_cidr    = var.vpc_cidr
  public_subnet_cidr = var.public_subnet_cidr
  region      = var.region
  availability_zones = var.availability_zones
}

module "security-groups" {
  source = "./security-groups"
  vpc_id    = module.network.vpc_id
  vpc_cidr  = var.vpc_cidr
  public_subnet_ids = module.network.pub_sub_ids
}
```

```
# ./network/vpc.tf
resource "aws_vpc" "default" {
  cidr_block           = var.vpc_cidr
  enable_dns_hostnames = true
}

output "vpc_id" {
  value = "${aws_vpc.default.id}"
}

resource "aws_subnet" "subnet_public" {
  vpc_id            = aws_vpc.default.id
  cidr_block        = var.public_subnet_cidr
  availability_zone = var.availability_zones
}

output "pub_sub_ids" {
  value = [ "${aws_subnet.subnet_public.*.id}" ]
}
```

Network



AWS

Route 53 DNS, TLS/SSL Certs, Regions, Availability Zones, VPC, Internet Gateway, Public Subnet, Private Subnet, Route Table, Network ACL, Direct

Azure

VNet, Network Gateway, NAT Gateway, Route Table, Express Route (on-prem), Public IP, Application Gateway

GCP

VPC, Subnet, Cloud NAT, Compute Route, Cloud Interconnect (on-prem), Public IP, API Gateway

VMware

Infoblox DNS / BIND, Verisign / Microsoft AD / Cloud Foundry CA TLS/SSL certs, Regions, Availability Zones, VLAN, Palo Alto / Checkpoint Firewall, DMZ, Internal VLANs, Cisco / Juniper / HP / Dell Route Table, Network ACL, WAN Link / Dark fiber, VMware ESXi / Tanzu NSX Firewall Rules, VMware vLAN

Security



AWS

AWS Config (resource), AWS GuardDuty (NIDS), AWS Macie (S3), VPC Flow Logs

Azure

Azure PolicySets, Network Security Groups, Azure AD Policies

GCP

GCP Security Command Center

VMware

Palo Alto Prisma (resource), Splunk (NIDS), SFlow / NetFlow / Cisco Network Flow Logs, Qualys / Tenable Nessus / Rapid7 Nexpose / Checkpoint (VM, container), Tripwire / OSSEC (FIM)

Identity



AWS

IAM Group, IAM Role, IAM User, IAM Policy
(customer-managed)

Azure

Azure AD (Active Directory), Azure Resource
Group

GCP

Service Account, Folder, Roles, Policy

VMware

Microsoft Active Directory, LDAP, SAML,
Okta

Monitoring



AWS

AWS CloudTrail (cli/sdk), CloudWatch, CloudWatch Metrics

Azure

Azure Network Watcher Flow Log, Monitor

GCP

Network Telemetry, VPC Flow Logs, Cloud Audit Logs

VMware

DataDog / SignalFX / Nagios / SolarWinds, Splunk / ELK / SumoLogic, HP OpenView

Compute



AWS

Load Balancer (ALB, ELB, NLB),
Auto-scaling Group + Launch Config +
Resource Group + EC2, EKS (K8S), ECS,
FarGate (hosted ECS), AWS Lambda

Azure

Traffic Manager (global LB), Scale Set +
Launch Config + Resource Group + VM,
Azure K8S / AKS

GCP

Load Balancer, Managed Instance Group
(MIG) + Instance Template + Stateful
Configuration + Compute, GCP EKS / K8S

VMware

F5 / HAProxy / nginx Load Balancers,
VMware vRealize, VMware Pivotal Cloud
Foundry (PKS, PCS) / K8S

Filestore



AWS

S3, CloudFront (CDN)

Azure

Blob Storage, Content Delivery Network

GCP

Cloud Storage, Cloud CDN

VMware

SAN, NAS, GlusterFS, Minio / Ceph / Dell
EMC ECS S3-compatible, Akamai

SQL Databases



AWS

RDS (MySQL, Aurora, Postgresql, MSSQL, Oracle)

Azure

MSSQL, Oracle, MySQL, Postgres

GCP

Cloud SQL (PostgreSQL, MySQL, SQL Server)

VMware

MS SQL Server, Oracle DB, Sybase DB, DB2, MySQL, Postgresql

NoSQL Databases



AWS

ElasticSearch, MongoDB, DocumentDB,
Hadoop, DynamoDB

Azure

ElasticSearch, MongoDB, Azure HDInsight
Hadoop

GCP

BigQuery, ElasticSearch, MongoDB Atlas,
BigTable

VMware

ElasticSearch, MongoDB, Hadoop

In-memory Databases



AWS

ElastiCache (Memcached, Redis)

Azure

Azure Cache for Redis

GCP

GCP Memorystore (Redis, Memcached)

VMware

Memcached, Redis



Private Module Registry

Private Module Registry (PMR)



Terraform Cloud's private module registry works similarly to the [public Terraform Registry](#) and helps you share [Terraform modules](#) across your organization. It includes support for module versioning, a searchable and filterable list of available modules, and a [configuration designer](#) to help you build new workspaces faster.

You can add both private and public modules to the registry:

- [Private modules](#) are hosted on the registry and are only available to members of that organization. In Terraform Enterprise, they are also available to other organizations that are configured to [share modules](#) with that organization.
- [Public modules](#) are automatically synchronized from the Terraform Registry where they are hosted. Public modules are not supported in Terraform Enterprise.

Private Module Registry (PMR)



Resources

- <https://learn.hashicorp.com/tutorials/terraform/module-private-registry-add?in=terraform/modules>
- <https://learn.hashicorp.com/tutorials/terraform/module-private-registry-share?in=terraform%2Fmodules>
- <https://www.terraform.io/docs/registry/index.html>
- <https://www.terraform.io/docs/cloud/registry/publish.html>
- <https://www.terraform.io/docs/cloud/registry/add.html>

The image features a dark blue background with decorative geometric patterns. In the top-left corner, there are several overlapping squares and rectangles filled with a fine grid of small white dots. Some of these shapes are further defined by thin, parallel white lines. In the bottom-right corner, there is a large rectangular area also filled with a fine grid of small white dots. The word "Workspaces" is prominently displayed in the center-left area in a bold, white, sans-serif font.

Workspaces

Considerations



- **Blast-Radius:** Do not put everything in one place.
- **Least Privilege:** Divide cloud resources into multiple Workspaces so that a Team cannot change another Team's cloud resources.
- **Rate of Change:** The Networking layer will not change as often as the Compute layer. Common changes should not affect uncommonly changing resources.
- **Ease of Maintenance:** Group similar resources to ensure maintenance changes don't affect other components, ex: upgrading all instances of Postgres / MySQL / MS SQL / Oracle / ElasticSearch should not affect the Networking resource.

1. Monolithic Workspace



<u>Production</u>	<u>Staging</u>	<u>QA</u>	<u>Dev</u>
network	network	network	network
security	security	security	security
identity	identity	identity	identity
compute	compute	compute	compute
filestore	filestore	filestore	filestore
sql	sql	sql	sql

2. Production vs. Non-production



<u>Production</u>	<u>Staging</u>	<u>QA</u>	<u>Dev</u>
network	network	network	network
security	security	security	security
identity	identity	identity	identity
compute	compute	compute	compute
filestore	filestore	filestore	filestore
sql	sql	sql	sql

3. Prod vs. Non-prod w/ Landing Zones



<u>Production</u>	<u>Staging</u>	<u>QA</u>	<u>Dev</u>
network security identity	network security identity	network security identity	network security identity
compute filestore sql	compute filestore sql	compute filestore sql	compute filestore sql

4. Divided by Environments (Envs)



Production

network
security
identity
compute
filestore
sql

Staging

network
security
identity
compute
filestore
sql

QA

network
security
identity
compute
filestore
sql

Dev

network
security
identity
compute
filestore
sql

5. Isolated Envs w/ Landing Zones (LZs)

<u>Production</u>	<u>Staging</u>	<u>QA</u>	<u>Dev</u>
network security identity	network security identity	network security identity	network security identity
compute filestore sql	compute filestore sql	compute filestore sql	compute filestore sql

6. Isolated Envs w/ LZs and App Layers

<u>Production</u>	<u>Staging</u>	<u>QA</u>	<u>Dev</u>
network security identity	network security identity	network security identity	network security identity
compute	compute	compute	compute
filestore	filestore	filestore	filestore
sql	sql	sql	sql

7. Isolated Envs w/ Shared App Layers



<u>Production</u>	<u>Staging</u>	<u>QA</u>	<u>Dev</u>
network	network	network	network
security	security	security	security
identity	identity	identity	identity
compute filestore sql	compute filestore sql	compute filestore sql	compute filestore sql

8. Isolated Envs w/ Isolated Layers



<u>Production</u>	<u>Staging</u>	<u>QA</u>	<u>Dev</u>
network	network	network	network
security	security	security	security
identity	identity	identity	identity
compute	compute	compute	compute
filestore	filestore	filestore	filestore
sql	sql	sql	sql

Workspace Creation Automation



```
# Configure a TF Workspace Variable called
# "tf_token" with the TFC API Token
terraform {
  required_providers {
    tfe = {
      source = "hashicorp/tfe"
      version = "~> 0.25.3"
    }
    null = {
      source = "hashicorp/null"
      version = "~> 3.1.0"
    }
  }
}
# https://registry.terraform.io/providers/hashicorp/tfe/latest/docs
provider "tfe" {
  hostname = var.tf_hostname
  token    = var.tf_token
}
```

Workspace Creation Automation



```
variable "tf_organization" {
  type = string
  default = "Pyrocumulus"
}

variable "tf_workspaces" {
  type = set(string)
  default = ["workspaceA", "workspaceB",
    "workspaceC"]
}

resource "tfe_workspace" "test" {
  for_each = var.tf_workspaces
  name = each.key
  organization = var.tf_organization
}

output "tf_workspace_ids" {
  value = { for k, v in tfe_workspace.test :
    k => v.id }
}
```

```
resource "tfe_variable" "test" {
  for_each = { for k, v in
tfe_workspace.test:
  k => v.id }
  key = "test_key_name"
  value = "test_value_name"
  category = "terraform"
  workspace_id = each.value
}

resource "tfe_team" "test" {
  name = "test-team-name"
  organization = var.tf_organization
}

resource "tfe_team_access" "test" {
  for_each = { for k, v in
tfe_workspace.test:
  k => v.id }
  access = "read"
  team_id = tfe_team.test.id
  workspace_id = each.value
}
```



Workspace Variables

Workspaces, Secrets / Credentials



1. Vault Enterprise
2. Vault Open Source
3. Cloud Agents, with Cloud Identity Credentials (ex: AWS IAM Instance Profile)
4. Variable Sets
5. **terraform_remote_state** data source, read between Workspaces
6. Workspace Variable, Sensitive
7. Workspace Environment Variable, Sensitive
8. CI/CD Inject Credentials at Run-time

<https://www.hashicorp.com/blog/managing-credentials-in-terraform-cloud-and-enterprise>



Git Repository Structure

Linus Torvalds Quote

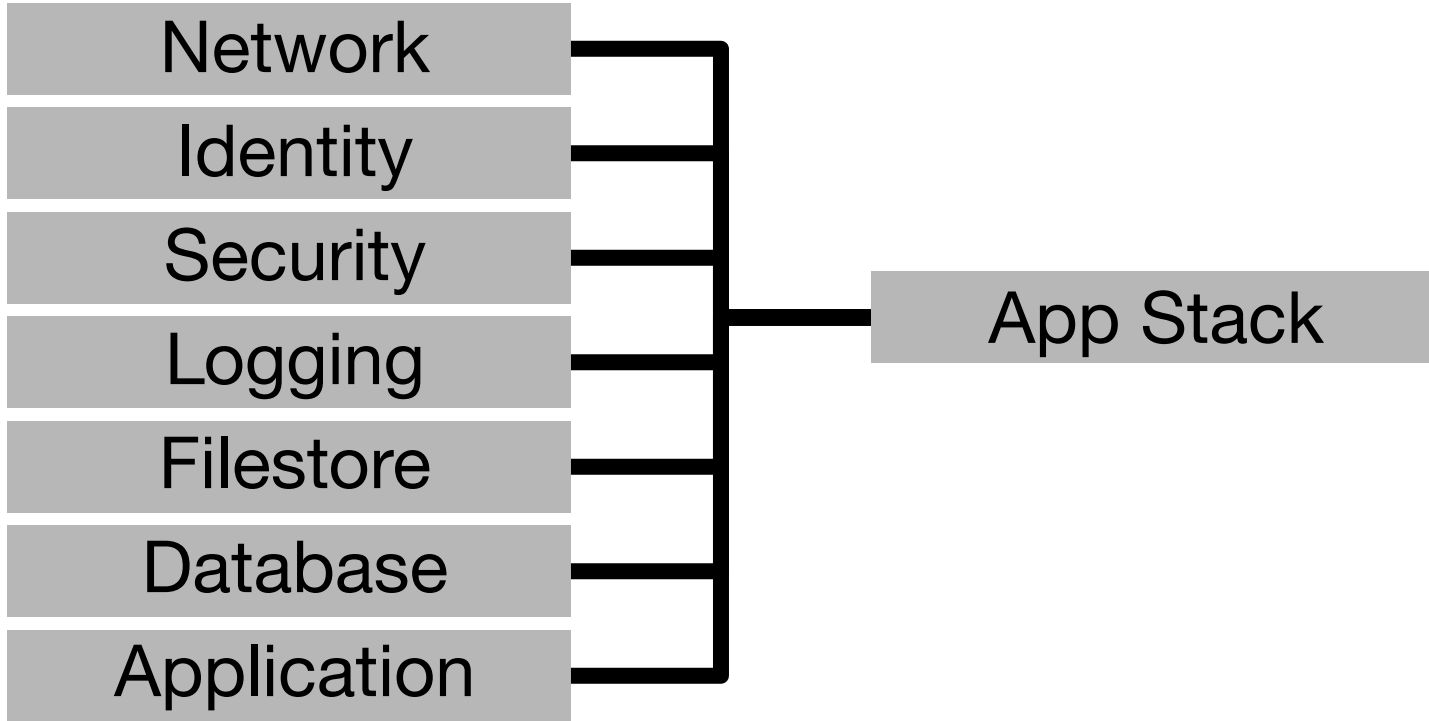


“So git scales really badly if you force it to look at everything as one `_huge_` repository. I don't think that part is really fixable, although we can probably improve on it.”

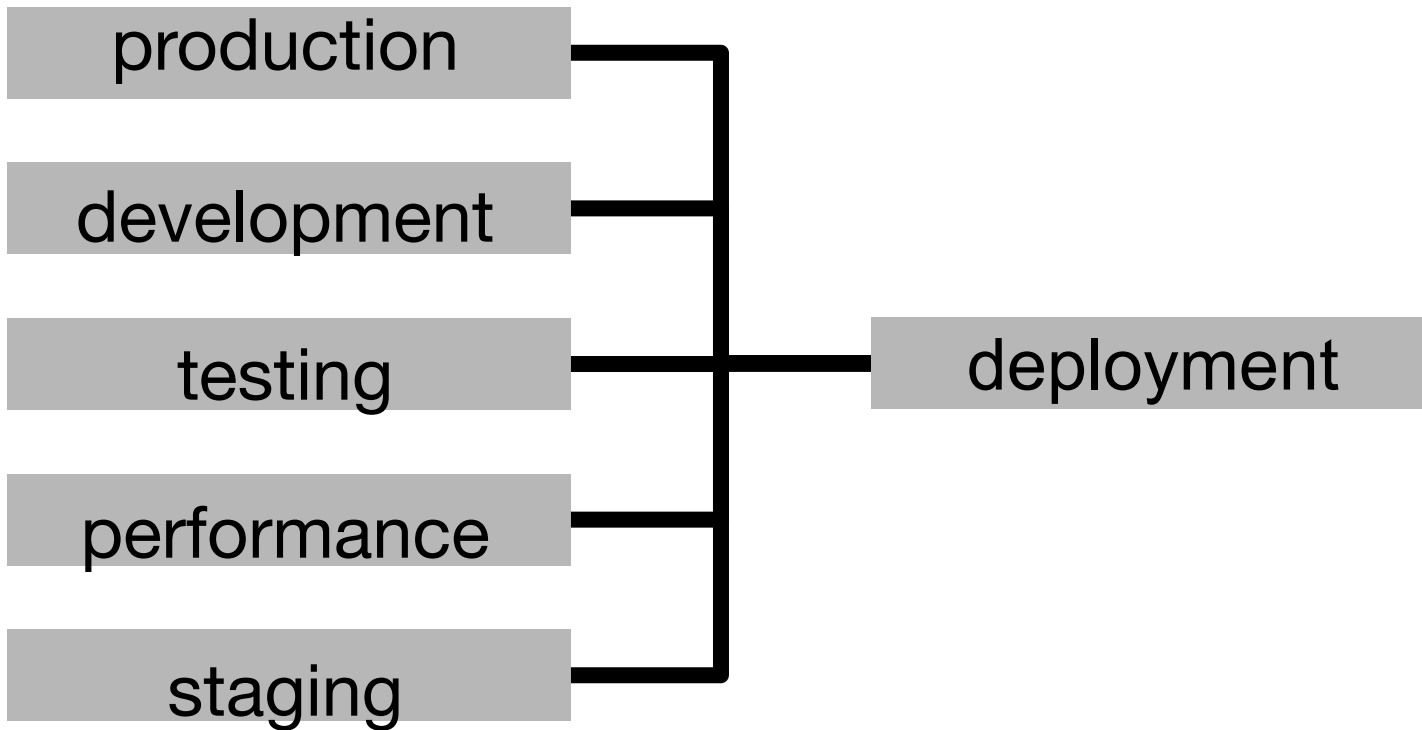
- **Linus Torvalds** (2009-05-01), Creator of Linux and Git

<https://marc.info/?l=git&m=124121401124923&w=2>

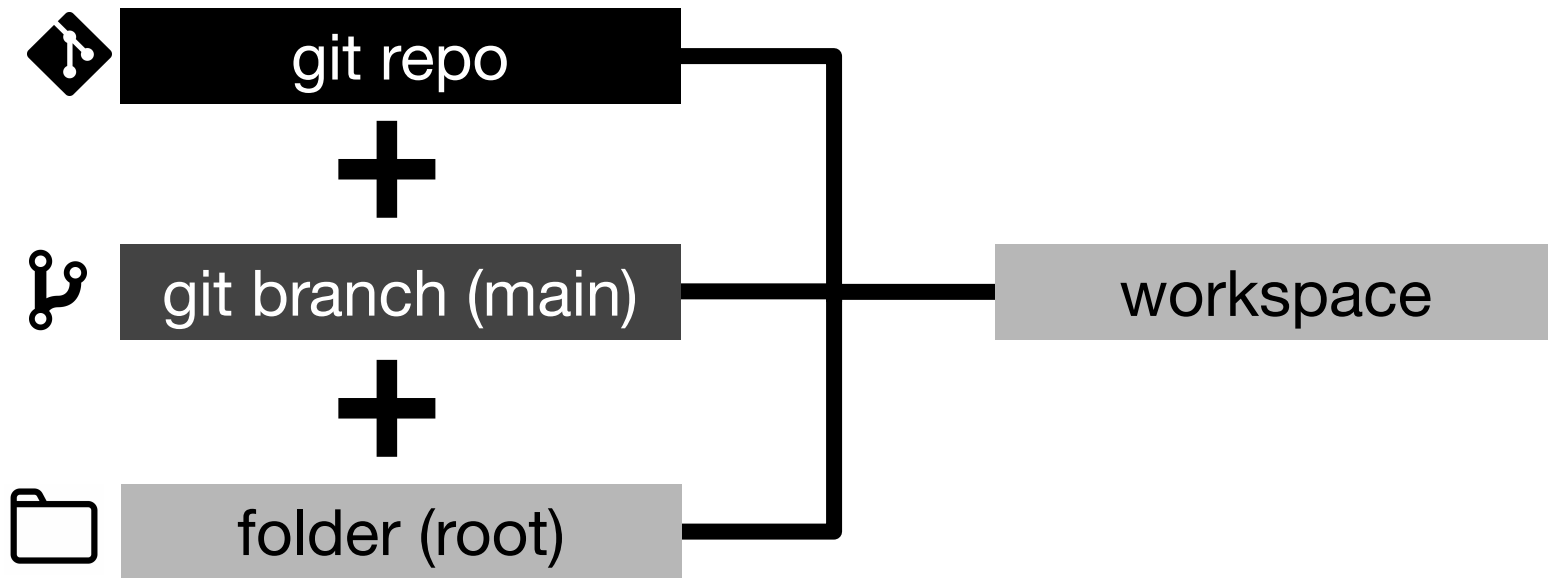
Shared Modules



Environments



Components, for VCS-Driven Runs



Monorepo, One Branch, One Folder



allApps-tf



main



terraform

- Workspace per Env
- One folder for all Envs
- impractical
- any errors would take down all infrastructure

Monorepo, One Branch, Many Folders



allApps-tf



main



tf-prod



tf-dev

- Workspace per Env
- Folder per Env
- one git repo
- one git branch
- large git clones
- duplicate code
- difficult git PR merges
- cannot git tag / version

Monorepo, Many Branches, One Folder



allApps-tf



production



terraform



dev



terraform

- Workspace per Env
- Git Branch per Env
- many git branches
- no duplicate code
- one git repo
- large git clones
- difficult git PR merges

Many Repos, One Branch, One Folder



appA-tf-prod



main



terraform



appA-tf-dev



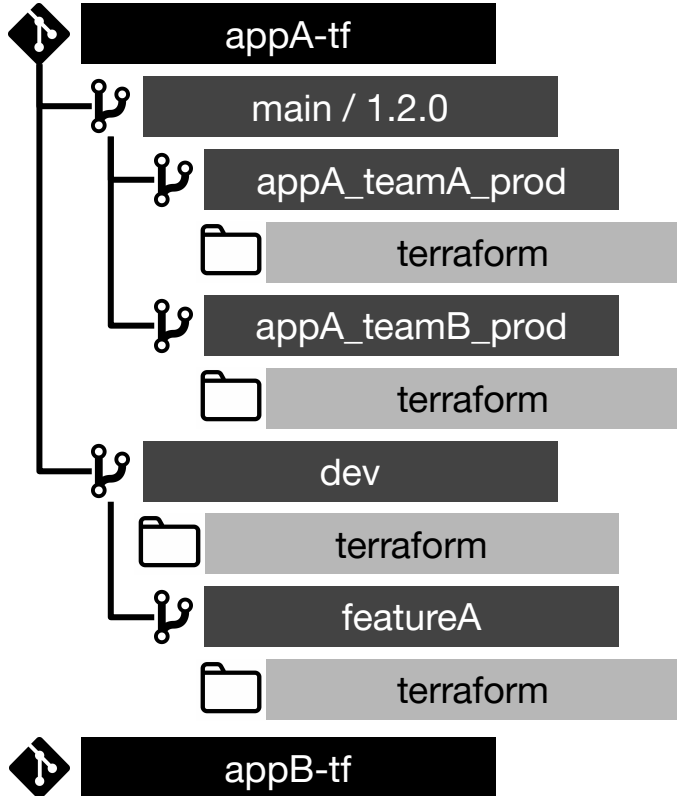
main



terraform

- Workspace per Env
- Git Repo per Env
- many git repos
- small git clones
- easy git PR merges
- one Folder per Env
- one git branch per repo
- duplicate code
- can't tag / version

Many Repos, Many Branches, One Folder



- Workspace per Env
- Git Branch per Env
- many Git Repos
- many Git Branches
- small git clones
- easy git PR merges
- no duplicate Code
- easily git tag / version



Refactor a Git Monorepo?

1. Refactor to use Terraform Modules
2. Create Git Repos for each Terraform Module
3. Created Git Repos of Terraform Code for each App
4. Use Git Branches for each App Instance, and each Environment (production, dev, etc.)
5. Use folders for the TF Modules / “App Layers”
6. Use a GitOps Workflow, using Git Branches, and a Terraform “VCS-Driven Workflow” to update and modify code and configuration changes.



Q & A



Thank You

customer.success@hashicorp.com

www.hashicorp.com/customer-success