Terraform the World

IaC Provisioner

One Infrastructure at a Time

About me!

Daniel Colón

https://www.linkedin.com/in/danielecolon/

A+, Security+, MCSE: Cloud Platform & Infrastructure

CompTIA A+

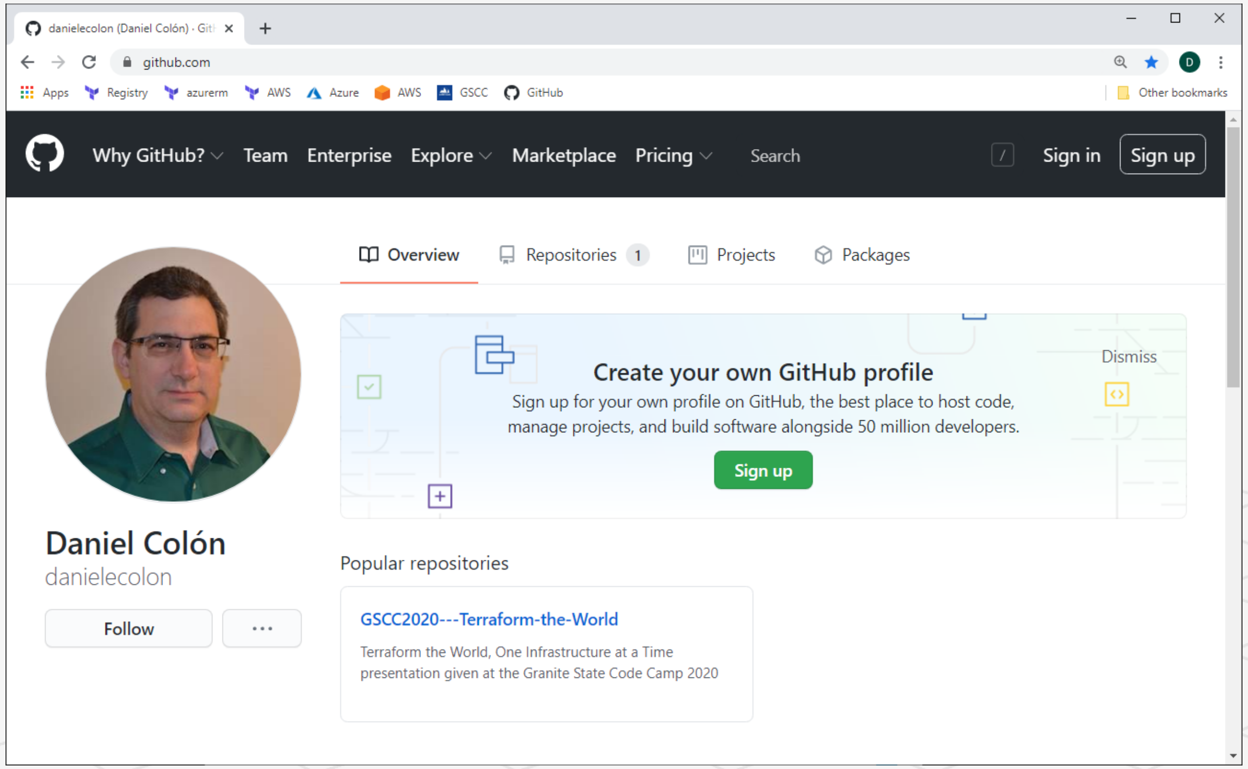
CompTIA Security+

Microsoft CERTIFIED AZURE SOLUTIONS ARCHITECT EXPERT

Slides/Code Samples

Slides

https://github.com/danielecolon


danielecolon (Daniel Colón) · Git

github.com

Apps Registry azurerm AWS Azure AWS GSCC GitHub Other bookmarks

Why GitHub? Team Enterprise Explore Marketplace Pricing Search Sign in Sign up

Overview Repositories 1 Projects Packages

Create your own GitHub profile

Sign up for your own profile on GitHub, the best place to host code, manage projects, and build software alongside 50 million developers.

Sign up

Dismiss

Daniel Colón

danielecolon

Follow

Popular repositories

GSCC2020---Terraform-the-World

Terraform the World, One Infrastructure at a Time presentation given at the Granite State Code Camp 2020

Agenda

Terraform Overview

Terraform Basics

Best Practices

Q&A

Terraform Overview

Tool for building, changing, and versioning infrastructure

Cloud agnostic

Open source written in GoLang

Uses HCL (Hashicorp Configuration Language)

Based on Infrastructure as Code Paradigm

Benefits of Infrastructure as Code

Software Development Methodologies

Version control, Modular Development, Testing

Agility

Automate deployment and recovery processes

Focus on Mean Time to Recovery

CI/CD – Continuously test, integrate, and deploy

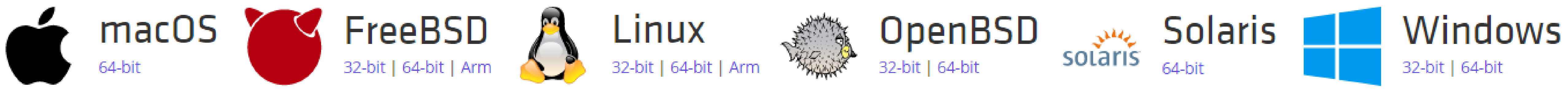
Make infrastructure immutable (when possible)

Avoids configuration drift

Terraform - Cloud Agnostic

Supports all Major Cloud Providers
https://www.terraform.io/docs/providers/type/major-index.html

Installation


macOS 64-bit
FreeBSD 32-bit | 64-bit | Arm
Linux 32-bit | 64-bit | Arm
OpenBSD 32-bit | 64-bit
Solaris 64-bit
Windows 32-bit | 64-bit

1. Download Terraform

   https://www.terraform.io/downloads.html

2. Set path to Terraform

Testing your installation Demo

Provider Authentication

Different providers have different methods of authentication

Some ways are more secure than others

Avoid authentication methods requiring hard coded secrets in terraform files

- Hard coded secrets are at risk of leaking into source control

AWS/Azure/GCP Authentication

AWS Provider

https://www.terraform.io/docs/providers/aws/index.html

Azure Provider

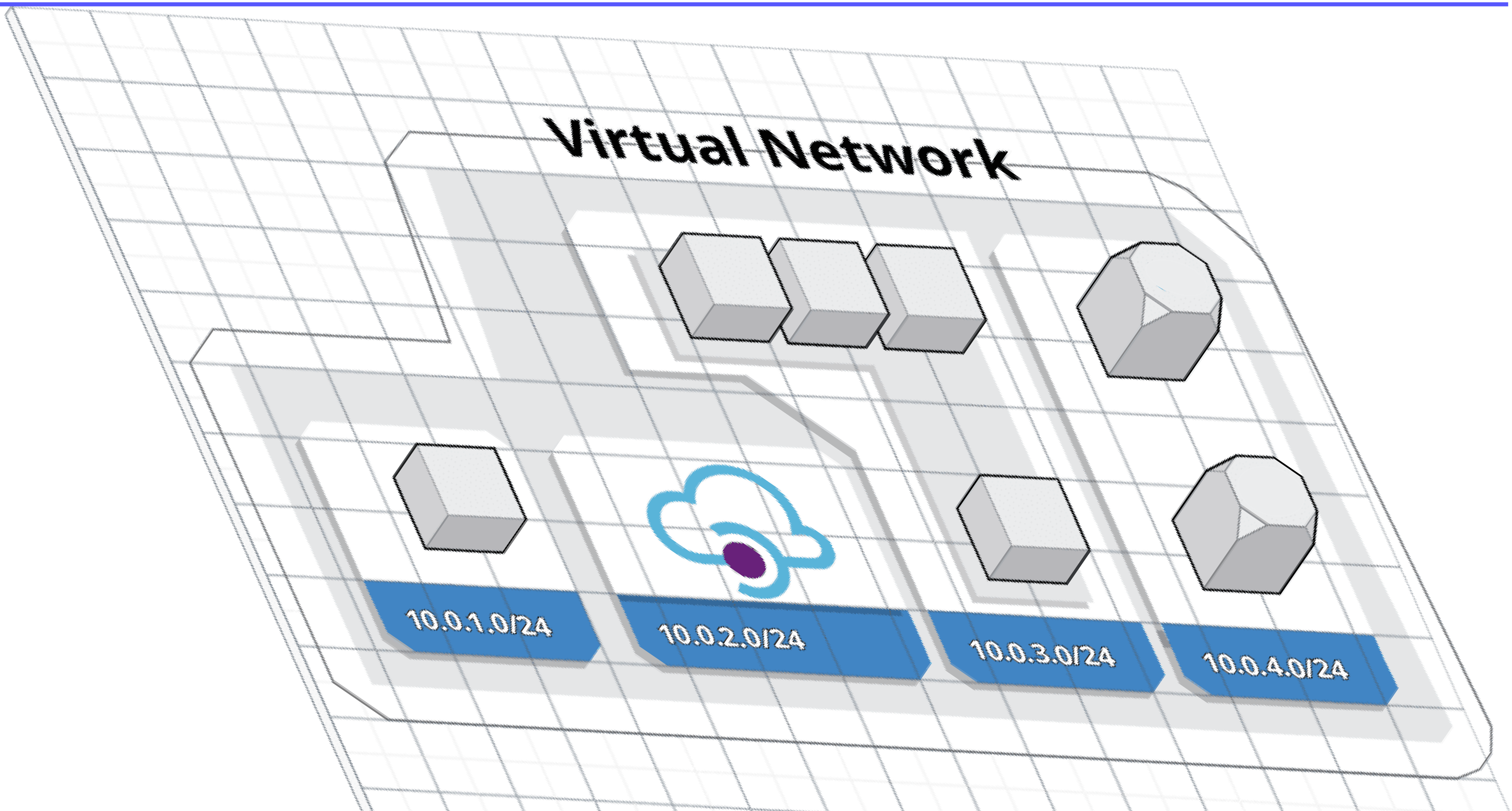
https://www.terraform.io/docs/providers/azurerm

GCP Provider

https://www.terraform.io/docs/providers/google/index.html

Authentication Demo

Infrastructure



Specify Terraform & Provider version

main.tf

```
# Configure Terraform
terraform {
  // (October 17, 2019)
  required_version = "=0.12.11"
}

# Configure the Azure Provider
provider "azurerm" {
  // (October 04, 2019)
  version = "=1.35.0"
}


...
```

Infrastructure as Code

main.tf

```
# Configure the Azure Provider
provider "azurerm" {
  version  = "=2.36.0"
  features {}
}
```

Infrastructure as Code

main.tf

```
# Create a resource group
resource "azurerm_resource_group" "rg" {
  name     = "vNet-rg"
  location = "East US"
}

# Create a virtual network
resource "azurerm_virtual_network" "vNet" {
  name                = "Dev-vNet"
  address_space       = ["10.0.0.0/16"]
  resource_group_name = azurerm_resource_group.rg.name
  location            = azurerm_resource_group.rg.location
}
```

Infrastructure as Code

main.tf

```
# Create a resource group
resource "azurerm_resource_group" "rg" {
  name     = "vNet-rg"
  location = "East US"
}

# Create a virtual network
resource "azurerm_virtual_network" "vNet" {
  name                = "Dev-vNet"
  address_space       = ["10.0.0.0/16"]
  resource_group_name = azurerm_resource_group.rg.name
  location            = azurerm_resource_group.rg.location
}
```

Terraform Lifecycle

1. Creation

```
terraform init
```

2. Plan

```
terraform plan -out terraform.tfplan
```

3. Apply

```
terraform apply terraform.tfplan
```

4. Decommission

```
terraform destroy
```

Terraform init

First command to run for a new configuration

Initializes various local settings and data

Terraform uses a plugin-based architecture

- Automatically download any Provider binary for providers used in configuration

Terraform plan

Used to create an execution plan

Determines actions to achieve desired state

Usage: terraform plan [options] [dir]
Example

```
terraform plan -out terraform.tfplan
```

Note: Plan files encode configuration, state, diff, and variables.
Variables might contain secrets.

Terraform apply

Applies changes required to reach desired state

Determines actions to achieve desired state

Usage: terraform apply [options] [dir-or-plan]
Example

```
terraform apply terraform.tfplan
```

Terraform destroy

Destroy the Terraform-managed infrastructure

Usage: terraform destroy [options] [dir]
Example

```
terraform destroy
```

Terraform Lifecycle Demo

Multiple Environments

1 Creation

```
terraform init
```

2 Plan

```
terraform plan -var-file=dev.tfvars -out terraform.tfplan
```

3 Apply

```
terraform apply -var-file=dev.tfvars terraform.tfplan
```

4 Decommission

```
terraform destroy
```

Use Variables

main.tf

```
# Create a resource group
resource "azurerm_resource_group" "rg" {
  name     = var.resource_group_name
  location = var.resource_group_location
}

# Create a virtual network
resource "azurerm_virtual_network" "vNet" {
  name                = var.virtual_network_name
  address_space       = var.virtual_network_address_space
  resource_group_name = azurerm_resource_group.rg.name
  location            = azurerm_resource_group.rg.location
}
```

Multiple Environments Demo

Data Sources

```
# Use data source to get info of existing Virtual Network
data "azurerm_virtual_network" "vNet" {
  name     = "WebVMDemo-rg"
  location = "East US"
}

# Here we are outputting the Virtual Network Id
output "virtual_network_id" {
  value = data.azurerm_virtual_network.vNet.id
}
```

Modules

Self-contained packages of Terraform configurations

A module can call other modules

Modules can be called multiple times

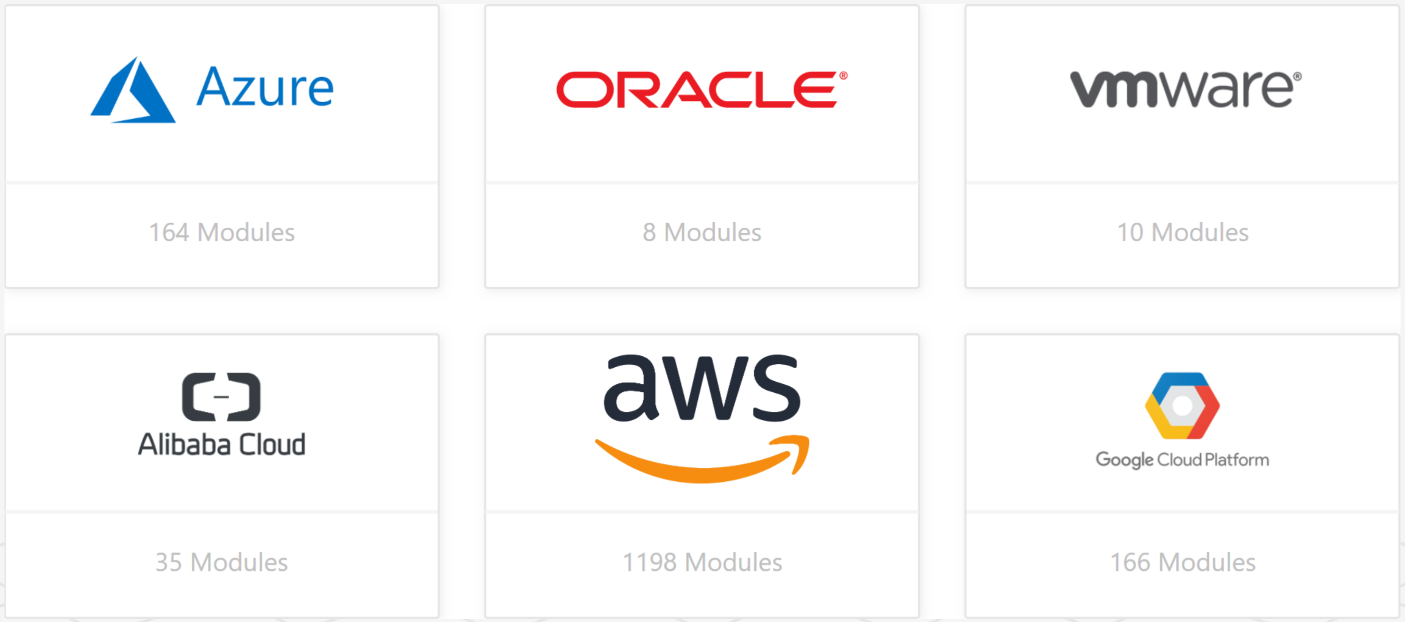
```
\modules.terraform\AWS\SecurityGroup

README.md
main.tf
var.tf
output.tf
```

Modules Demo

Terraform Registry

Discover providers for any service, and modules for common infrastructure configurations

https://registry.terraform.io/



State

Stored in local file “terraform.tfstate” by default

Maps resources to your configuration

Used to create plans and make changes to infrastructure

Remote State

Terraform writes state data to a remote data store

This can be shared between all members of a team

Remote state is a feature of backends

Azure can store remote state in a storage blob

AWS can store remote state in S3

GCP can store remote state in a bucket

Locking

Prevents two users from writing to remote state at the same time

Must be supported by backend to work

Terraform Cloud supports this locking feature

- Access to shared state and secret data
- Control for approving infrastructure changes
- Private registry for modules
- See https://www.terraform.io/docs/cloud/index.html for more details

Source Control

modules.tf

```
# Create a Network
module "network" {
  source= "=github.com/Azure/terraform-azurerm-network"

  location = "east us"
}
```

Versioning

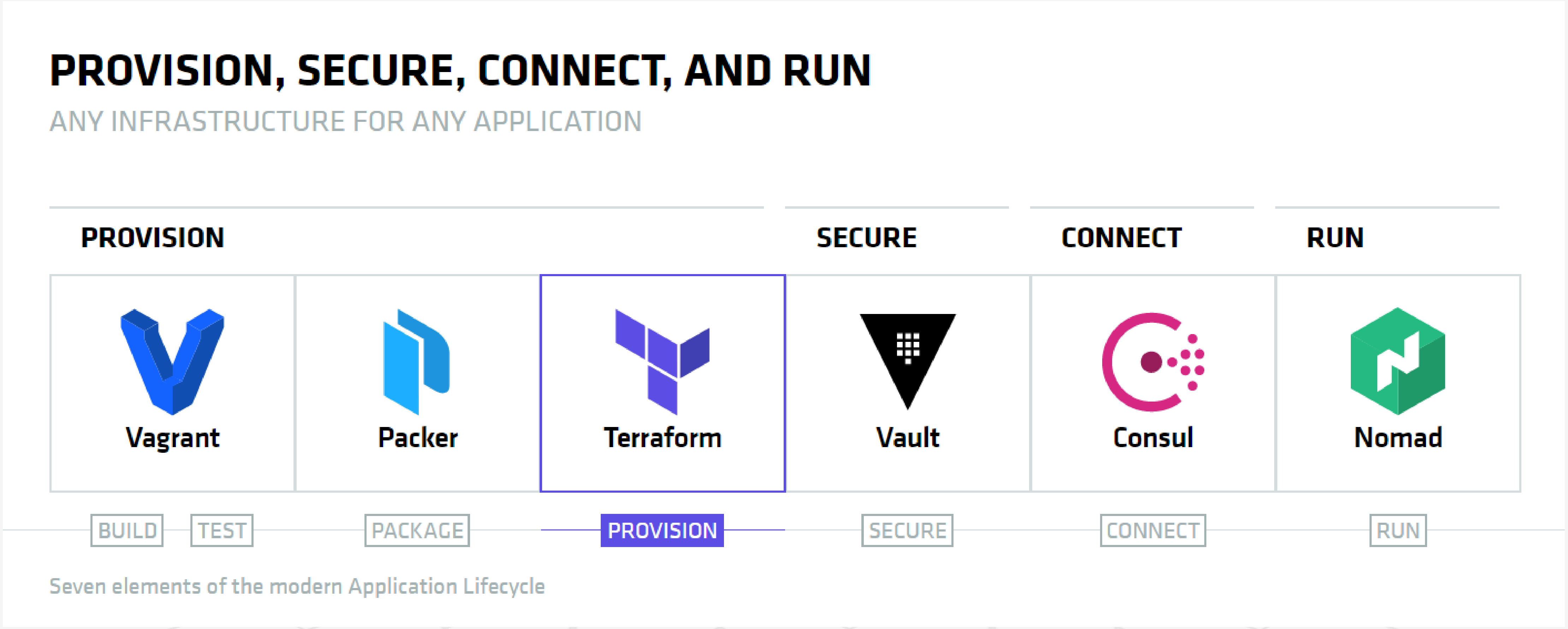
modules.tf

```
# Create a Network
module "network" {
 source= "github.com/Azure/terraform-azurerm-network?ref=v2.0.0"

 location = "east us"
}
```

Best Practices

- Use small reusable modules
- Use Remote State
- Use versioning
  - Specify the version for your provider
  - Specify the version of terraform
  - Use a separate repo for modules and tagging to specify version
- Hard code nothing!

What's Next



PROVISION, SECURE, CONNECT, AND RUN

ANY INFRASTRUCTURE FOR ANY APPLICATION

Seven elements of the modern Application Lifecycle

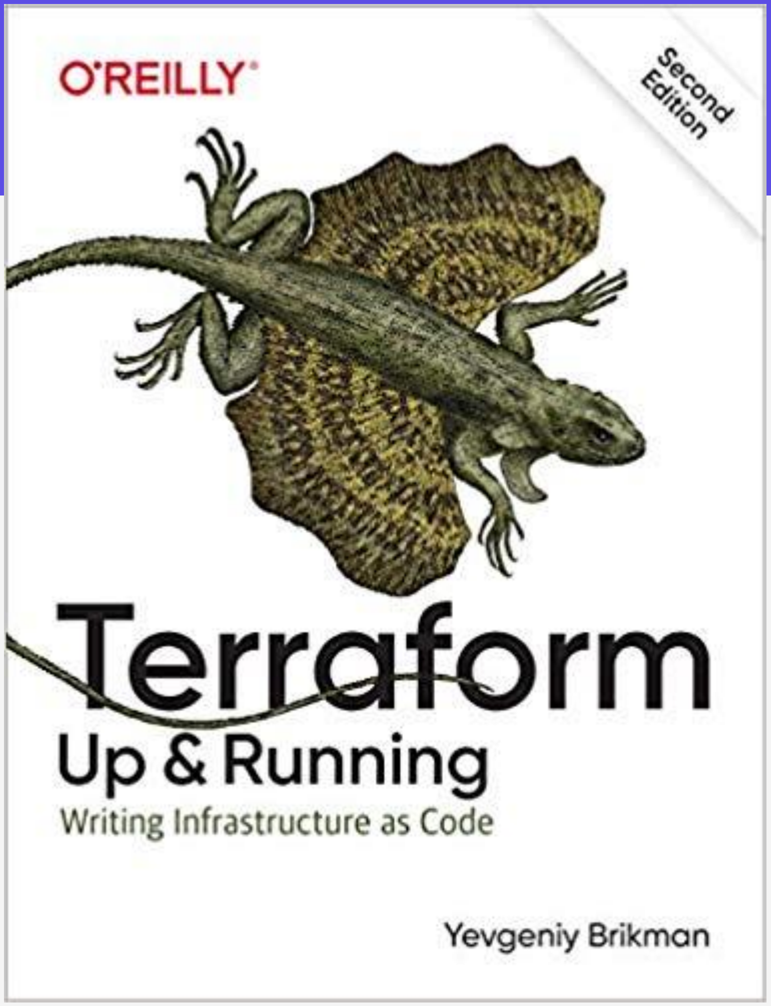
What's Next

Terraform	Use infrastructure as code to consistently provision any cloud, infrastructure, and service
Vault	Manage secrets and protect sensitive data
Consul	Automate service-based networking in the cloud
Nomad	Deploy and manage containerized, legacy, or batch application
Vagrant	Development Environments made easy
Packer	Build Automated Machine Images
Sentinel	Policy as code framework for HashiCorp Enterprise Products

More questions about Terraform?

Terraform Home Page
https://www.terraform.io

Getting Started
https://learn.hashicorp.com/terraform/



Recommend Reading: Terraform Up & Running by Yevgeniy Brikman

Summary

- "Go out and Terraform the world one infrastructure at a time."

Resources

Terraform Documentation

- https://www.terraform.io/docs/index.html

Terraform Registry

- https://registry.terraform.io/browse/providers

Slides

- https://github.com/danielecolon

Next

Time	Concord
1:00 PM	**Troubleshooting Performance Issues in SharePoint Online Sites** **Sean McDonough** Microsoft MVP & Lover of All Things Donut
2:10 PM	**What does Azure Synapse mean for my modern data warehouse?** **Chris Seferlis** Transforming Business by Architecting Cloud and Data Solutions that immediately provide value
3:20 PM	**Real-Time content using Azure SignalR Service** **Udaiappa Ramachandran** Architect, Cloud Expert
4:30 PM	**Getting started with Blazor WebAssembly** **Nathan Westfall** Software Engineering Team Lead at Tyler Technologies
5:30 PM	**Closing is in the Manchester Virtual Room**