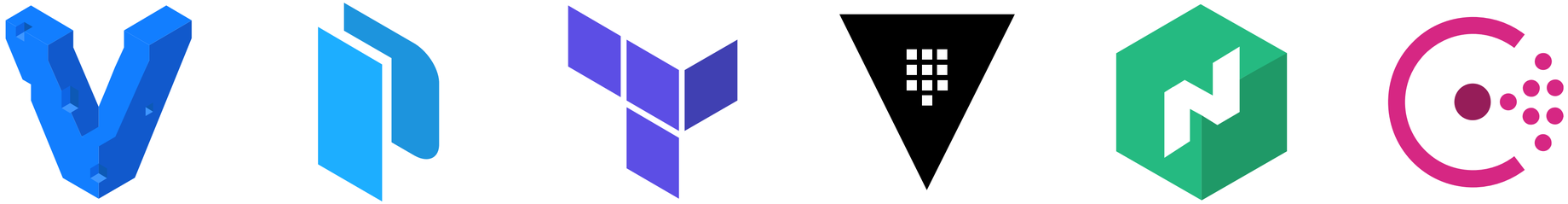
Beyond "Hello World": Working with Terraform in a Team

Tom Harvey, Terraform Engineer
@tombuildsstuff

Copyright © 2018 HashiCorp



Agenda

- The Terraform Statefile
- Interacting with existing resources
- Bringing existing resources into Terraform
- The Terraform & Provider Blocks
- Reusing configurations with Modules

Copyright © 2018 HashiCorp

The Terraform Statefile

Keeping track of the state of the world

Copyright © 2018 HashiCorp

The Terraform Statefile

How does Terraform keep track of the state of the world?

- Terraform tracks the state of any resources it's created in a Statefile
  - Every resource e.g. Resource Groups / VPC's

Copyright © 2018 HashiCorp

The Terraform Statefile

How does Terraform keep track of the state of the world?

- Terraform tracks the state of any resources it's created in a Statefile
  - Every resource e.g. Resource Groups / VPC's
- By default this lives on your local disk - but can be stored remotely in a Backend on:
  - Artifactory
  - AWS
  - Azure
  - Google
  - Terraform Enterprise
  - (and more)

Copyright © 2018 HashiCorp

The Terraform Statefile

Configuring a Backend (AWS)

```
$ cat backend.tf

terraform {
  backend "s3" {
    bucket = "foo"
    key    = "bar"
    region = "us-east-1"
  }
}

$ terraform init


...
```

Copyright © 2018 HashiCorp

The Terraform Statefile

Configuring a Backend (Azure)

```
$ cat backend.tf

terraform {
  backend "azurerm" {
    storage_account_name = "abcd1234"
    container_name       = "tfstate"
    key                  = "prod.terraform.tfstate"
  }
}

$ terraform init

...
```

Copyright © 2018 HashiCorp

Working with existing resources

Interacting with existing resources & bringing existing resources into Terraform

Copyright © 2018 HashiCorp

Working with existing resources

- Interacting with Existing Resources
  - Data Sources allow you to reference existing resources by name

Copyright © 2018 HashiCorp

Data Sources

Accessing information about existing resources

```
$ cat resource_group.tf

data "azurerm_resource_group" "existing" {
  name = "my-existing-resource-group"
}

output "location" {
  value = "${data.azurerm_resource_group.existing.location}"
}
```

Copyright © 2018 HashiCorp

Working with existing resources

- Interacting with Existing Resources
  - Data Sources allow you to reference existing resources by name
  - Remote State allows you to access the outputs and properties of another Terraform Statefile

Copyright © 2018 HashiCorp

Remote State

Accessing information in another Statefile

```
$ cat first/main.tf

terraform {
  backend "s3" {
    bucket = "foo"
    key    = "bar"
    region = "us-east-1"
  }
}

resource "aws_subnet" "database" {
  # ...
}

output "subnet_id" {
  value = "${aws_subnet.database.id}"
}
```

Copyright © 2018 HashiCorp

Remote State

Accessing information in another Statefile

```
$ cat second/backend.tf

data "terraform_remote_state" "network" {
  backend = "s3"
  config {
    bucket = "terraform-state-prod"
    key    = "network/terraform.tfstate"
    region = "us-east-1"
  }
}

resource "aws_instance" "foo" {
  # ...
  subnet_id = "${data.terraform_remote_state.vpc.subnet_id}"
}
```

Copyright © 2018 HashiCorp

Working with existing resources

- Interacting with Existing Resources
  - Data Sources allow you to reference existing resources by name
  - Remote State allows you to access the outputs and properties of another Terraform Statefile
- Bring existing resources into Terraform
  - Importing resources via `terraform import`

Copyright © 2018 HashiCorp

Importing Resources

Importing an existing Resource into Terraform

```
$ cat resource.tf

resource "aws_vpc" "test" {
  cidr_block = "10.0.0.0/16"
}
```

Copyright © 2018 HashiCorp

Importing Resources

Importing an existing Resource into Terraform

```
$ terraform import aws_vpc.test vpc-a01106c2
aws_vpc.test: Importing from ID "vpc-a01106c2"...
aws_vpc.test: Import complete!
  Imported aws_vpc (ID: vpc-a01106c2)
aws_vpc.test: Refreshing state... (ID: vpc-a01106c2)

Import successful!

The resources that were imported are shown above. These
resources are now in
your Terraform state and will henceforth be managed by
Terraform.
```

Copyright © 2018 HashiCorp

The Terraform/Provider Blocks

Pinning versions, backends and more

Copyright © 2018 HashiCorp

Terraform/ Provider Blocks

The Terraform & Provider blocks

- Terraform Block
  - Specify which version of Terraform is required
  - Backend configuration

Copyright © 2018 HashiCorp

The Terraform Block

Required version of Terraform and Backends

```
terraform {
  required_version = ">= 0.11.7"
}
```

Copyright © 2018 HashiCorp

The Terraform Block

Required version of Terraform and Backends

```
terraform {
  required_version = ">= 0.11.7"

  backend "s3" {
    bucket = "foo"
    key    = "bar"
    region = "us-east-1"
  }
}
```

Copyright © 2018 HashiCorp

Terraform/ Provider Blocks

The Terraform & Provider blocks

- Terraform Block
  - Specify which version of Terraform is required
  - Backend configuration
- The Provider Block
  - Pinning a version of a Provider
  - Working with multiple instances

Copyright © 2018 HashiCorp

Provider Blocks

Pinning a provider version

```
provider "azurerm" {
  version = "=1.4.0"
}
```

Copyright © 2018 HashiCorp

Provider Blocks

Using multiple versions of a Provider

```
provider "azurerm" {
  subscription_id = "00000000-0000-0000-000000000000"
  alias           = "production"
}

provider "azurerm" {
  subscription_id = "11111111-1111-1111-111111111111"
  alias           = "staging"
}
```

Copyright © 2018 HashiCorp

Provider Blocks

Using multiple versions of a Provider

```
provider "azurerm" {
  subscription_id = "00000000-0000-0000-000000000000"
  alias           = "production"
}

provider "azurerm" {
  subscription_id = "11111111-1111-1111-111111111111"
  alias           = "staging"
}

data "azurerm_virtual_network_gateway" "production" {
  # ...
  provider = "azurerm.production"
}

data "azurerm_virtual_network_gateway" "staging" {
  # ...
  provider = "azurerm.staging"
}
```

Copyright © 2018 HashiCorp

Modules

For when resources are basically the same

Copyright © 2018 HashiCorp

Terraform Modules

A collection of reusable Terraform configurations

- Terraform supports Modules, which are a collection of reusable configurations
- Data Sources, Resources, Variables and Outputs can be used as normal in Modules
- Can be sourced from Local Disk, other Git Repositories or Module Registries

Copyright © 2018 HashiCorp

Custom Modules

An example of a Terraform Configuration

```
$ cat modules/resource-group/main.tf

variable "prefix" {}
variable "location" {}

resource "azurerm_resource_group" "test" {
  name     = "${var.prefix}-resources"
  location = "${var.location}"
}
```

Copyright © 2018 HashiCorp

Custom Modules

Using a custom module in the local disk

```
$ cat main.tf

module "from-local-disk" {
  source   = "./modules/resource-group"
  prefix   = "devopsinpractice"
  location = "UK South"
}
```

Copyright © 2018 HashiCorp

Custom Modules

Using a custom module from another Git repository

```
$ cat main.tf

module "from-a-git-repo" {
  source   = "github.com/my-org/azurerm-resource-group"
  prefix   = "devopsinpractice"
  location = "UK South"
}
```

Copyright © 2018 HashiCorp

Custom Modules

Using a custom module from a sub-directory within a different Git repository

```
$ cat main.tf

module "from-a-sub-directory" {
  source   = "github.com/my-org/azurerm-resource-group//
some-directory"
  prefix   = "devopsinpractice"
  location = "UK South"
}
```

Copyright © 2018 HashiCorp

registry.terraform.io

Terraform | Module Registry

Browse Publish Sign-in

Terraform Module Registry

Discover modules for common infrastructure configurations for any provider

Popular searches: vault, aws, database

Find Terraform Modules

Use and learn from verified and community modules

POPULAR MODULES

aws security-group aws

aws elb aws

Copyright © 2018 HashiCorp

Registry Modules

Using a module from the Terraform Module Registry

```
$ cat main.tf

module "rg" {
  source   = "Azure/network-security-group/azurerm"
  version  = "1.0.1"
  location = "West Europe"
}
```

Copyright © 2018 HashiCorp

HashiDays

AMSTERDAM

June 25-27, 2018

HashiConf '18

October 22-24, 2018 | San Francisco

Resources

- Terraform Website - terraform.io
- Module Registry - registry.terraform.io
- HashiDays Amsterdam - hashidays.com
- HashiConf (San Francisco) - hashiconf.com

- Repo: github.com/tombuildsstuff/devops-in-practice

Copyright © 2018 HashiCorp

Thank you.

HashiCorp

www.hashicorp.com hello@hashicorp.com