Overview

- Exploring functions and the CLI
- Exploring Backend
- Workspaces

Functions

- Built in to terraform
- Function_name(arg1,arg2,arg3)
- Test in terraform console
- Several type of functions

Common Function Categories

Numeric min(42, 13, 7)	**String** lower("TACOS")	**Collection** merge(map1, map2)
Filesystem file(path)	**IP network** cidrsubnet()	**Date and time** timestamp()

Terraform Functions

```
#Configure networking

variable network_info {

 default = "10.1.0.0/16" #type, default, description

}

#Returns 10.1.0.0/24

cidr_block = cidrsubnet(var.network_info, 8, 0)

#Returns 10.1.0.5

host_ip = cidrhost(var.network_info,5)
```

Terraform Functions

```
#Create ami map
variable "amis" {
  type = "map"
  default = {
    us-east-1 = "ami-1234"
    us-west-1 = "ami-5678"
  }
}
ami = lookup(var.amis, "us-east-1", "error")
```

Terraform Providers

IaaS, PaaS, and SaaS

Community and HashiCorp

Open source

Resources and data sources

Multiple instances

Provider Example

```
provider “azurerm” {
  subscription_id = "subscription-id"
  client_id = "principal-used-for-access"
  client_secret = "password-of-principal"
  tenant_id = "tenant-id"
  alias = "arm-1"
}
resource "azurerm_resource_group" "azure_tacos" {
  name = "resource-group-name"
  location = "East US"
  provider = azurerm.arm-1
}
```

Resource Arguments

depends_on

count

for_each

provider

Depends_on and Count

```
resource "aws_instance" "taco_servers" {
  count = 2
  tags {
    Name = "customer-${count.index}"
  }
  depends_on = [aws_iam_role_policy.allow_s3]
}
```

For_each

```
resource "aws_s3_bucket" "taco_toppings" {

  for_each = {

    food = "public-read"
    cash = "private"

  }

  bucket = "${each.key}-${var.bucket_suffix}"

  acl = each.value

}
```

Backend

- A "backend" in Terraform determines how state is loaded and how an operation such as apply is executed.
- This abstraction enables non-local file state storage, remote execution, etc.
- By default, Terraform uses the "local" backend, which is the normal behavior of Terraform you're used to
- https://www.terraform.io/docs/backends/types/index.html

Benefits of Backends

- **Working in a team**: Backends can store their state remotely and protect that state with locks to prevent corruption. Some backends such as Terraform Cloud even automatically store a history of all state revisions.
- **Keeping sensitive information off disk**: State is retrieved from backends on demand and only stored in memory. If you're using a backend such as Amazon S3, the only location the state ever is persisted is in S3.
- **Remote operations**: For larger infrastructures or certain changes, terraform apply can take a long, long time. Some backends support remote operations which enable the operation to execute remotely. You can then turn off your computer and your operation will still complete. Paired with remote state storage and locking above, this also helps in team environments.

Workspace

- The persistent data stored in the backend belongs to a *workspace*
- Initially the backend has only one workspace, called "default", and thus there is only one Terraform state associated with that configuration.
- Certain backends support *multiple* named workspaces, allowing multiple states to be associated with a single configuration
- https://www.terraform.io/docs/state/workspaces.html