Module 3 Terraform Providers





Agenda

- Introduction to Providers
- Local Provider
- Public Cloud Providers
- AWS resource examples
- Azure resource examples
- Google Cloud resource examples
- Lab 3





Introduction to Providers

- A provider in Terraform is a plugin that enables interaction with an API. The providers are specified in the Terraform configuration code. They tell Terraform which services it needs to interact with.
- Terraform currently has 1700+ providers
- Each provider adds a set of resource types and/or data sources that Terraform can manage.



Local Provider

- The Local provider is used to manage local resources, such as files.
- Terraform primarily deals with remote resources which are able to outlive a single Terraform run, and so local resources can sometimes violate its assumptions.
- Local resources are best used with care, since depending on local state can make it hard to apply the same Terraform configuration on many different local systems where the local resources may not be universally available

```
resource "local_file" "foo" {
  content = "foo!"
  filename = "${path.module}/foo.bar"
}
```

```
data "local_file" "foo" {
   filename = "${path.module}/foo.bar"
}

resource "aws_s3_object" "shared_zip" {
   bucket = "my-bucket"
   key = "my-key"
   content = data.local_file.foo.content
}
```



Public Cloud Providers

- Terraform can provision infrastructure across public cloud providers such as Amazon Web Services (AWS), Azure and Google Cloud, as well as private cloud and virtualization platforms such as OpenStack and VMWare.
- Providers differ! Always refer to documentation regarding terminology and resource types and features

What cloud providers does Terraform support?

Providers

- AWS.
- Azure.
- · Google Cloud Platform.
- Kubernetes.
- · Alibaba Cloud.
- Oracle Cloud Infrastructure.





Aliases

 You can optionally define multiple configurations for the same provider, and select which one to use on a per-resource or per-module basis. The primary reason for this is to support multiple regions for a cloud platform; other examples include targeting multiple Docker hosts, multiple Consul hosts, etc.

```
# The default provider configuration; resources that begin with `aws_` will u
# it as the default, and it can be referenced as `aws`.

provider "aws" {
    region = "us-east-1"
}

# Additional provider configuration for west coast region; resources can
# reference this as `aws.west`.

provider "aws" {
    alias = "west"
    region = "us-west-2"
}
```





AWS resource example

Here is a sample deployment of a VPC with subnets, routing tables and routing table associations.

Note: The Internet Gateway, NAT Gateway and Security group resources have been omitted for brevity

```
vpc_id = aws_vpc.prod.ic
```





```
esource "aws_vpc" "prod" {
    cidr_block = "10.1.0.0/16"
 resource "aws_subnet" "public" {
    vpc_id = aws_vpc.prod.id
    cidr_block = "10.1.1.0/24"
 resource "aws_subnet" "private" {
    vpc_id = aws_vpc.prod.id
    cidr_block = "10.1.2.0/24"
  tags = {
Name = "Private"
resource "aws_route_table_association" "a"
subnet_id = aws_subnet.public.id
route_table_id =
aws_route_table.public_rt.id
resource "aws_route_table_association" "b" {
    subnet_id = aws_subnet.private.id
    route_table_id =
    aws_route_table.private_rt.id
resource "aws_route_table" "public_rt" {
    vpc_id = aws_vpc.prod.id
    cidr_block = "0.0.0.0/0"
 gateway_id=aws_internet_gateway.myigw.id
resource "aws_route_table" "private_rt" {
    vpc_id = aws_vpc.prod.id
      cidr_block = "0.0.0.0/0"
 gateway_id=aws_nat_gateway.mynatgtw.id
```

```
resource "aws_vpc" "prod" {
 cidr_block = "10.1.0.0/16"
```

```
resource "aws_subnet" "public" {
 vpc_id = aws_vpc.prod.id
 cidr_block = "10.1.1.0/24"
  tags = {
  Name = "Public"
resource "aws_subnet" "private" {
 vpc_id = aws_vpc.prod.id
 cidr_block = "10.1.2.0/24"
  tags = {
  Name = "Private"
```

```
aws_route_table.public_rt.id
resource "aws_route_table_association" "b" {
 subnet_id = aws_subnet.private.id
 route_table_id =
aws_route_table.private_rt.id
```

route_table_id =

resource "aws_route_table_association" "a" {

subnet_id = aws_subnet.public.id

```
resource "aws_route_table" "public_rt" {
 vpc_id = aws_vpc.prod.id
    cidr_block = "0.0.0.0/0"
gateway_id=aws_internet_gateway.myigw.id
resource "aws_route_table" "private_rt" {
```

```
vpc_id = aws_vpc.prod.id
   cidr_block = "0.0.0.0/0"
gateway_id=aws_nat_gateway.mynatgtw.id
```



```
resource "aws_vpc" "prod" {
    cidr_block = "10.1.0.0/16"
}
```

```
resource "aws_subnet" "public" {
   vpc_id = aws_vpc.prod.id
   cidr_block = "10.1.1.0/24"

   tags = {
   Name = "Public"
  }
}
```

```
resource "aws_route_table_association" "a" {
  subnet_id = aws_subnet.public.id
  route_table_id = aws_route_table.public_rt.id
}
```

```
resource "aws_route_table" "public_rt" {
   vpc_id = aws_vpc.prod.id

route {
    cidr_block = "0.0.0.0/0"
    gateway_id=aws_internet_gateway.myigw.id
}
```

```
resource "aws_subnet" "private" {
   vpc_id = aws_vpc.prod.id
   cidr_block = "10.1.2.0/24"

   tags = {
    Name = "Private"
   }
}
```

```
resource "aws_route_table_association" "a" {
    subnet_id = aws_subnet.private.id
    route_table_id = aws_route_table.private_rt.id
}
```

```
resource "aws_route_table" "private_rt" {
   vpc_id = aws_vpc.prod.id

route {
    cidr_block = "0.0.0.0/0"
    gateway_id=aws_nat_gateway.mynatgtw.id
}
```



```
resource "aws_subnet" "public" {
                                                                                                                                      resource "aws_route_table" "public_rt" {
                                        vpc_id = aws vpc.prod.id
                                                                                                                                        vpc_id = aws_vpc.prod.id
                                        cidr block = "10.1.1.0/24"
                                                                                 resource "aws_route_table_association" "a" {
                                                                                   subnet_id = aws_subnet.public.id
                                                                                                                                        route {
                                                                                   route_table_id = aws_route_table.public_rt.id
                                        tags = {
                                                                                                                                          cidr_block = "0.0.0.0/0"
                                        Name = "Public"
                                                                                                                                          gateway_id=aws_internet_gateway.myigw.id
resource "aws_vpc" "prod" {
 cidr_block = "10.1.0.0/16"
                                      resource "aws_subnet" "private" {
                                                                                                                                      resource "aws_route_table" "private_rt" {
                                        vpc_id = aws vpc.prod.id
                                                                                                                                        vpc_id = aws_vpc.prod.id
                                        cidr block = "10.1.2.0/24"
                                                                                 resource "aws_route_table_association" "a" {
                                                                                   subnet_id = aws_subnet.private.id
                                                                                                                                        route {
                                                                                   route_table_id = aws_route_table.private_rt.id
                                        tags = {
                                                                                                                                          cidr block = "0.0.0.0/0"
                                        Name = "Private"
                                                                                                                                          gateway_id=aws_nat_gateway.mynatgtw.id
```



```
resource "aws_vpc" "prod" {
    cidr_block = "10.1.0.0/16"
}
```

```
resource "aws subnet" "public" {
   vpc_id = aws_vpc.prod.id
   cidr_block = "10.1.1.0/24"

   tags = {
   Name = "Public"
  }
}
```

```
resource "aws_route_table_association" "a" {
    subnet_id = aws_subnet.public.id
    route_table_id = aws_route_table.public_rt.id
}
```

```
resource "aws route table" "public rt"

vpc_id = aws_vpc.prod.id

route {
    cidr_block = "0.0.0.0/0"
    gateway_id=aws_internet_gateway.myigw.id
}
```

```
resource "aws subnet" "private" {
   vpc_id = aws_vpc.prod.id
   cidr_block = "10.1.2.0/24"

   tags = {
    Name = "Private"
   }
}
```

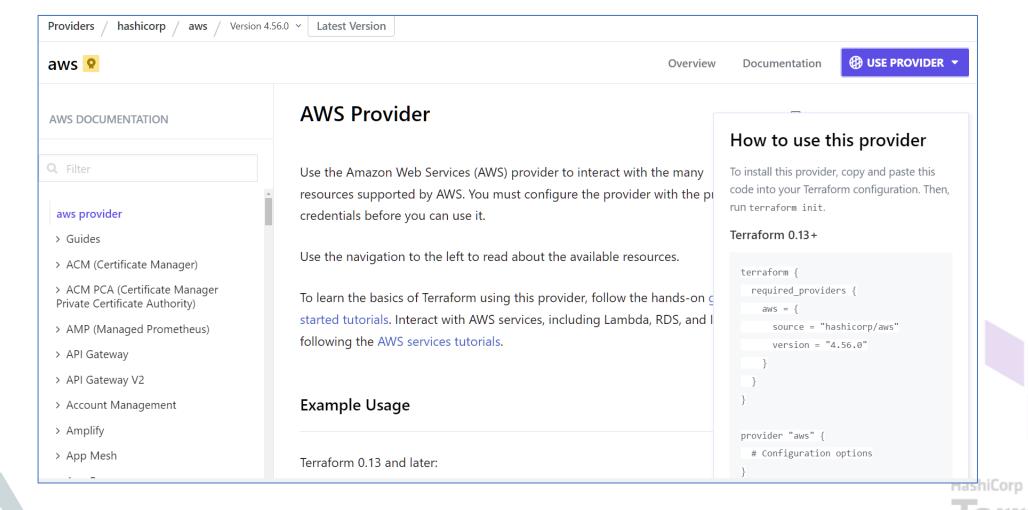
```
resource "aws_route_table_association" "a" {
    subnet_id = aws_subnet.private.id
    route_table_id = aws_route_table.private_rt.id
}
```

```
resource "aws route table" "private rt'
vpc_id = aws_vpc.prod.id

route {
    cidr_block = "0.0.0.0/0"
    gateway_id=aws_nat_gateway.mynatgtw.id
}
```



Terraform registry: aws



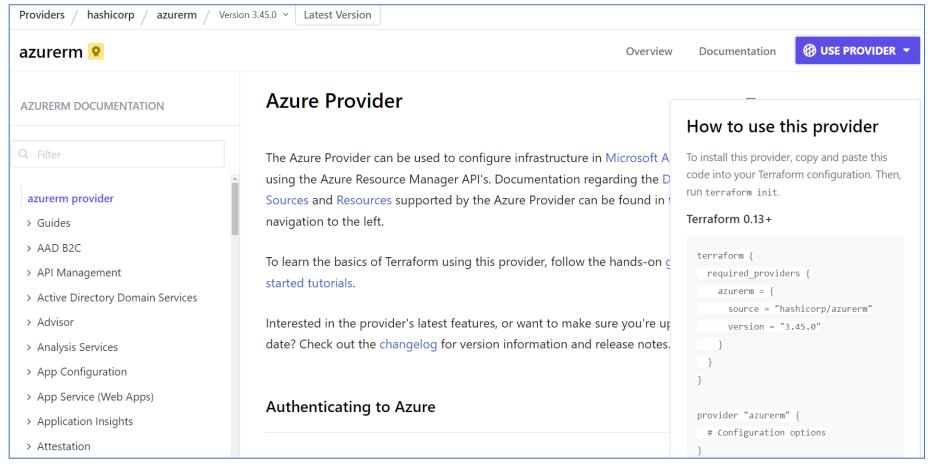
Azure resource example

An Azure Virtual Network

```
resource "azurerm_resource_group" "rg1" {
  name = "prod-resources"
  location = "West Europe"
}
```



Terraform registry: azurerm







Google Cloud resource example

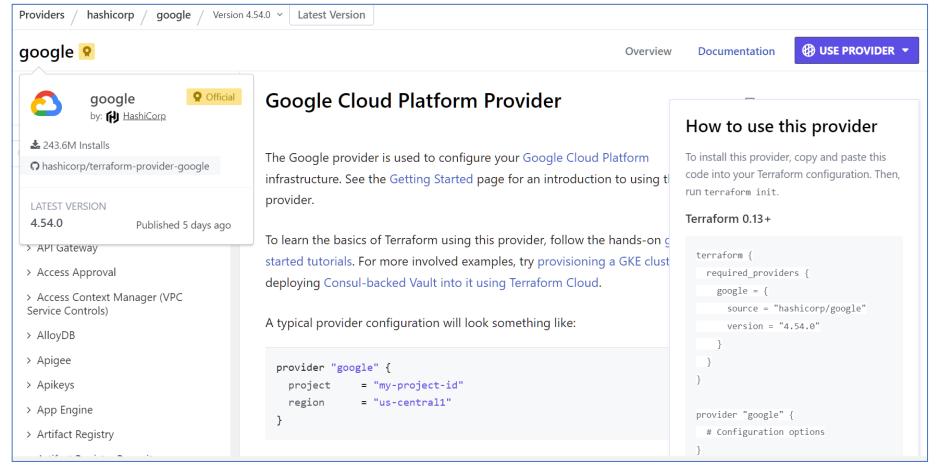
A Google Cloud VPC

```
resource "google_compute_network" "custom-test" {
    name = "test-network"
    auto_create_subnetworks = false
}
```





Terraform registry: google







Lab Group Discussion & Explore

Scoping

- What Type of Resources are you going to create?
- What specific Properties do we need to consider for these resources?

Workflow

- Are there any Implicit or Explicit dependencies?
- What TF Provider, Resource, and Modules are required?

Research

Identify the TF documentation that will assist in developing the solution





Lab 3

In this lab you will deploy a simple VPC architecture comprising of a VPC, subnets, routing tables, gateways and security groups.



Create a basic VPC deployment on AWS





Any questions...





