Terraform Data sources

Table of Content

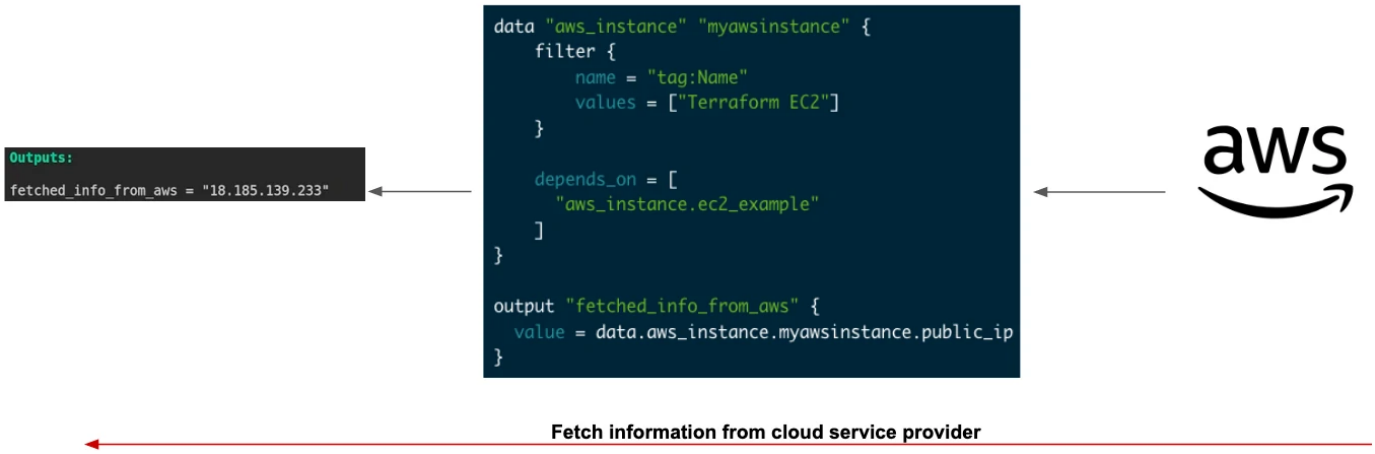
Terraform data sources can be beneficial if you want to retrieve or fetch the data from the cloud service providers such as AWS, AZURE, and GCF . Most of the time when we use Terraform along with AWS/AZURE/GCP then we always send data in terms of instructions or configuration.

But what if you want to get the information(arn, tags, owner_id, etc.) back from the cloud service provider AWS/AZURE/GCP?

Answer - We need to use the *data sources* to get the resource information back.

So *Terraform Data Sources* are a kind of an API that fetches the data/information from the resources running under the cloud infra and sends it back to terraform configuration for further use.

In this blog, we will look at the example in which we are going to create an `aws_instance` resource and then create a data source to fetch some of the information associated with the `aws_instance`.



1. Create an *aws_instance*

The motive of this exercise is to create an `aws_instance` and then create a *data source* to fetch all the possible Data Sources: aws_instance attributes.

Let's first write the terraform configuration for starting a `t2.mirco` aws_instance.

*(*Note- Replace the access_key and secret_key with your AWS account. Click here to know how to generate the access_key and secret_key)*

```
provider "aws" {
   region     = "eu-central-1"
   access_key = "AKIATQ37NXB2JMXVGYPG"
   secret_key = "ockvEN1DzYynDuKIh56BVQv/tMqmzvKnYB8FttSp"
}

resource "aws_instance" "ec2_example" {

   ami           = "ami-0767046d1677be5a0"
   instance_type =  "t2.micro"

   tags = {
           Name = "Terraform EC2"
   }
}
```

2. Define a data source

Now we have created our *aws_instance* in Step 1, let's add the data source to the existing terraform configuration.

Here is the data source configuration for fetching all the information of *aws_instance* -

```
data "aws_instance" "myawsinstance" {
    filter {
      name = "tag:Name"
      values = ["Terraform EC2"]
    }

    depends_on = [
      "aws_instance.ec2_example"
    ]
}
```

Key points to pay attention for -

1. **filter**: Although we have created only one instance still we have used filter because in a production-like environment you might have multiple *aws_instance* running, so you need to filter the instance anyhow. And since we have tagged our *aws_instance* with the name *Terr aform EC2* so we are going to use the same name inside the filter also.
2. **depends_on**: The second important parameter is *depends_on* because the `data source` does not know on its own which resource it belongs to, so we are going to add the *depends_on* parameter.

3. Create Output variable for data source

So far in Step 1 and Step 2, we have created the *aws_instance* and *data source*, now let's create an `output value` so that we can see all the information fetched or retrieved by the data source.

Here is the terraform configuration for the output value -

```
output "fetched_info_from_aws" {
  value = data.aws_instance.myawsinstance
}
```

Key points to pay attention for -

1. We have linked the output value to the data source which we have created in Step 2.

   To link the output value we are going to use the *data source* name. i.e. - `data.aws_instance.myawsinstance`

4. Apply the final terraform configuration along with *data source* and *output values*

Alright now I am assuming you have gone through all the 3 steps(Step 1,Step 2, and Step 3), so here is our final terraform configuration including *aws_instance, data source, and output values*

```
provider "aws" {
    region     = "eu-central-1"
    access_key = "AKIATQ37NXB2JMXVGYPG"
    secret_key = "ockvEN1DzYynDuKIh56BVQv/tMqmzvKnYB8FttSp"
}

resource "aws_instance" "ec2_example" {

    ami           = "ami-0767046d1677be5a0"
    instance_type =  "t2.micro"

    tags = {
      Name = "Terraform EC2"
    }
}

data "aws_instance" "myawsinstance" {
    filter {
        name = "tag:Name"
        values = ["Terraform EC2"]
    }

    depends_on = [
      "aws_instance.ec2_example"
    ]
}

output "fetched_info_from_aws" {
  value = data.aws_instance.myawsinstance.public_ip
}
```

You can simply run the following terraform command to create your *aws_instance* -

```
terraform init
```

BASH

```
terraform plan
```

BASH

```
terraform apply
```