Guide for using Packer + Terraform with Huawei Cloud.

Issue	**01**
Date	**2023-10-23**

Huawei Technologies Co., Ltd.

Huawei Technologies Co., Ltd.

Address:	Huawei Industrial Base Bantian, Longgang Shenzhen 518129 People's Republic of China
Website:	https://www.huawei.com/en/
Email:	support@huawei.com

Contents

1 Best Practices

1.1 Solution Overview

Applicable Scenarios

Deploying multiple ECS with the similar configurations can become a tedious, repetitive and long task.

- Manually deploying ECS through the console is a very repetitive task, and can become prone to human error while selecting specifications, and other configurations through the console.
- Using the console alone does not allow a task to be automated, so when more than one ECS needs to be created at a time it becomes a slow process.

This section describes and explains how to use the Huawei Cloud Packer plugin, along with Terraform in order to deploy ECS on Huawei Cloud.

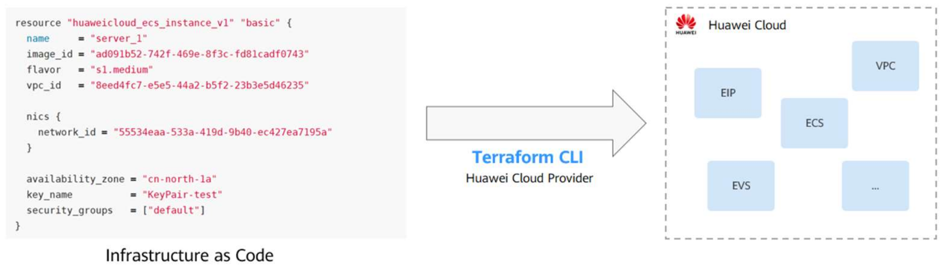
Architecture

This section describes the principles and practices of using packer templates and Terraform.

Packer is an open source tool for creating identical machine images for multiple platforms using a single source configuration. Packer is a lightweight, and runs on every major operating system, and is highly performant. Capable of creating machine images for multiple platform in parallel.

Packer **does not replace** configuration management tools, instead it can use them to install software onto the image.

According to the Packer Documentation a *machine image* is a single static unit that contains a pre-configured operating system and installed software which is used to quickly create new running machines.

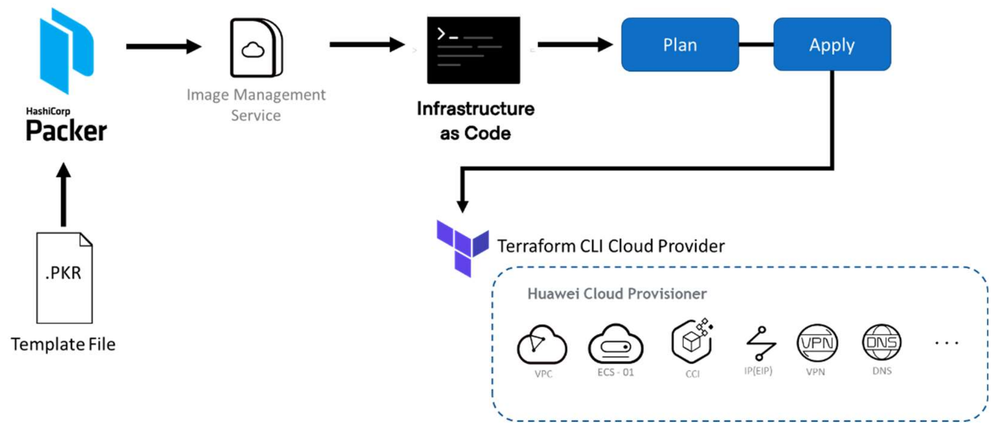


```
resource "huaweicloud_ecs_instance_v1" "basic" {
  name     = "server_1"
  image_id = "ad091b52-742f-469e-8f3c-fd81cadf0743"
  flavor   = "s1.medium"
  vpc_id   = "8eed4fc7-e5e5-44a2-b5f2-23b3e5d46235"

  nics {
    network_id = "55534eaa-533a-419d-9b40-ec427ea7195a"
  }

  availability_zone = "cn-north-1a"
  key_name          = "KeyPair-test"
  security_groups   = ["default"]
}
```

Figure 1. Using Terraform to manage cloud resources.

Terraform is an open-source tool used to orchestrate and manage IT infrastructure. You can describe an application or entire data center in configuration files. Terraform is used to easily create, manage, delete and version cloud resources on HUAWEI CLOUD.

In this guide we will base on the following architecture. First using Packer to create an image and then using the created image with Terraform to deploy ECS with code.



Highlights

- Lightweight, Opensource, multiplatform.
- Using a single configuration file for multiple images.
- Create ECS for multiple platforms in Parallel.

Constraints

- Terraform manages the state of the resources. If they get changed from other sources such as the Huawei Cloud console, this will make the configuration file not match reality. Which in turn makes management or modifying resources more challenging within Terraform

Resources and Prices

Table 1-1 Resources and prices

Region	Resource	Resource Name	Description	Quantity	Monthly Cost (USD)
LA-Mexico City2	Virtual Private Cloud (VPC)	Generated Automatically	CIDR block: 10.0.0.0/16	1	0.00
	VPC subnets	Generated Automatically	CIDR block: 10.0.1.0/24 Local subnet for establishing a Layer-2 connection to the remote subnet. This subnet is where the ECS is located.	1	0.00
	Elastic Cloud Server (ECS)	ecs-packer	Flavor: s6.large.2 System Disk: 40 GB	1	104.17
	Image Management Service	Ubuntu-2022-ProvidedByPacker	The image generated by packer which gets uploaded on IMS	1	0.00

The estimated resource cost is **104.17 USD**, excluding the fees generated during host migration. For details about the migration fees, see Pricing Details.

NOTICE

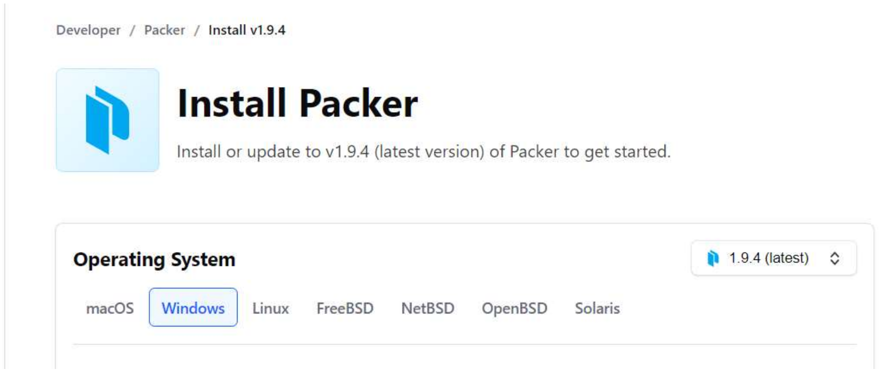
The prices provided here are for reference only. The actual prices are shown on the Huawei Cloud console.

1.2 Installing Packer on the local machine.

Either inside an ECS or a local machine you can download and install packer, access the download page of the product through this link. Hashicorp Packer Download.

As Packer is a multiplatform program, select the version appropriate to your OS. In this document will follow installation as if using a Windows Machine.

Step 1 Enter the download link.



Developer / Packer / Install v1.9.4

Install Packer

Install or update to v1.9.4 (latest version) of Packer to get started.

Operating System

1.9.4 (latest)

macOS [✓ Windows] Linux FreeBSD NetBSD OpenBSD Solaris

Step 2 Unzip the downloaded file onto a folder.

Organize the text by sections as follows:

1.3 Generating Template file for preparing image.

To create an image using Packer a template in the format **.pkr.hcl** format. Inside the template you need to configure builders, provisioners, and post-processors.

In providers, you can modify the source image as you need. For example, modify configurations and specify the software to be pre-installed.

In the GitHub repository of the plugin, you can find the required parameters for building your template file.

The resulting image from the packer template will be available in **Image Management Service** and will be used by terraform when creating multiple ECS in parallel.

Here is also an example template:

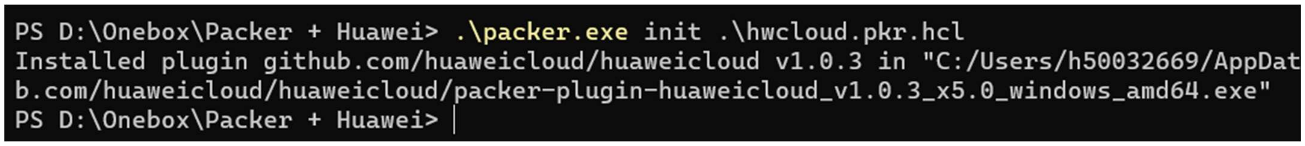
```
packer {
  required_plugins {
    huaweicloud = {
      version = ">= 1.0.0"
      source  = "github.com/huaweicloud/huaweicloud"
    }
  }
}

source "huaweicloud-ecs" "artifact" {
  region            = "xxx"
  availability_zone = "xxx"
  flavor            = "c6.large.2"
  source_image_name = "Ubuntu 22.04 server 64bit"
  image_name        = "Ubuntu-2204-image-powered-by-Packer"
  image_tags = {
    builder = "packer"
    os      = "Ubuntu-22.04-server"
  }

  ssh_username       = "root"
  eip_type           = "5_bgp"
  eip_bandwidth_size = 5
}

build {
  sources = ["source.huaweicloud-ecs.artifact"]

  provisioner "shell" {
    inline = ["apt-get update -y"]
  }

  post-processor "manifest" {
    strip_path = true
    output     = "packer-result.json"
  }
}
```

When using this template do not forget to set the following parameters: **region, availability zone**

1.4 Create the image using the template

If it is the **first time** running the packer template with the Huawei plugin, it is necessary to initialize the plugin first. This is done by running the following command.

```
$ packer init hwcloud.pkr.hcl
```



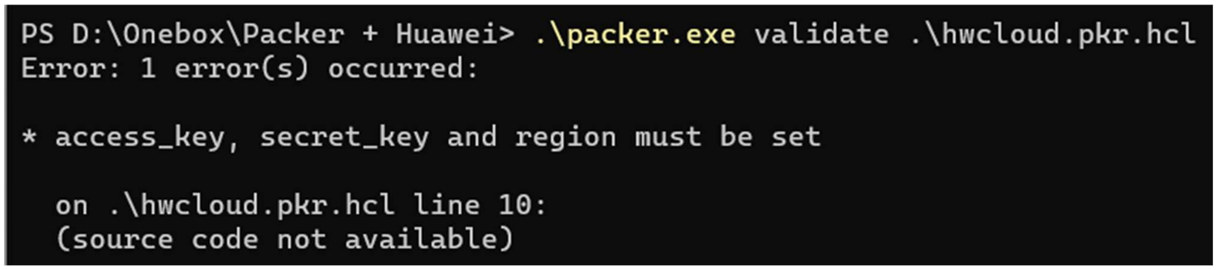
```
PS D:\Onebox\Packer + Huawei> .\packer.exe init .\hwcloud.pkr.hcl
Installed plugin github.com/huaweicloud/huaweicloud v1.0.3 in "C:/Users/h50032669/AppDat
b.com/huaweicloud/huaweicloud/packer-plugin-huaweicloud_v1.0.3_x5.0_windows_amd64.exe"
PS D:\Onebox\Packer + Huawei> |
```

The output of previous command should look like above.

Validate if your template does not have any errors or missing parameters by running the validate command

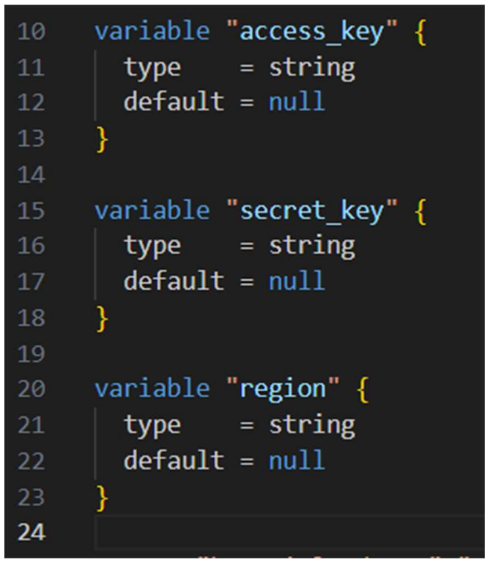
```
$ packer validate hwcloud.pkr.hcl
```

Running the command without changing anything on the template will generate the following error



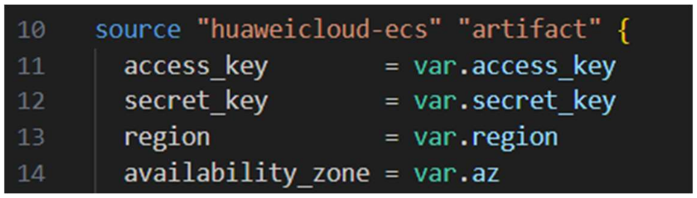
```
PS D:\Onebox\Packer + Huawei> .\packer.exe validate .\hwcloud.pkr.hcl
Error: 1 error(s) occurred:

* access_key, secret_key and region must be set

  on .\hwcloud.pkr.hcl line 10:
  (source code not available)
```

Modify the template to add the following lines below the packer field in the file.



```
variable "access_key" {
  type    = string
  default = null
}

variable "secret_key" {
  type    = string
  default = null
}

variable "region" {
  type    = string
  default = null
}

```

Also add the following lines below the source field in the file.



```
source "huaweicloud-ecs" "artifact" {
  access_key        = var.access_key
  secret_key        = var.secret_key
  region            = var.region
  availability_zone = var.az
```

1.4.1 Creating a Variable file for the template.

In order to fix the previous error, it is needed to specify your Access Key and Secret Access Key in the Packer Template script, but it is not a good practice to include the keys in the template as they will be stored as plaintext. Sharing this template will expose your AK/SK to the public where bad actors could use the keys to have access to your resources on the cloud.

The Variable file can look like this:

```
access_key = "TYPE YOUR AK HERE"
secret_key = "TYPE YOUR SK HERE"
region = "SPECIFY A REGION"       --ex:  la-south-2
```

You can check the region names and endpoints here.

A variable file is only to be used on locally in the machine that is going to run the build command and is formatted as in terraform.

1.4.2 Building the Template.

Now it will be possible to run the build command to generate the image by adding the **-var-file** flag.

Step 1 Execute the following command. To validate if the template is correctly configured.

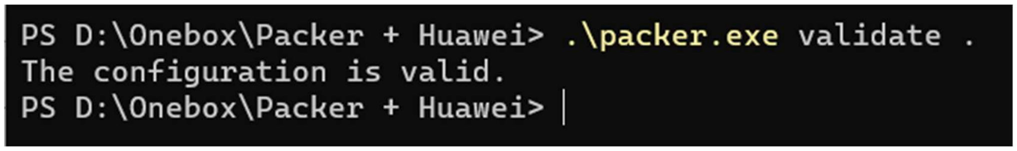
```
$ packer validate hwcloud.pkr.hcl -var-file=”variables.pkrvars.hcl”
```

You can avoid to pass the file via command line by renaming the file to anything that matches the expression: ***.auto.pkrvars.hcl**, then you can simply call with:

```
$ packer validate .
```

NOTE

Do not skip the “. “ at the end of the command.



```
PS D:\Onebox\Packer + Huawei> .\packer.exe validate .
The configuration is valid.
PS D:\Onebox\Packer + Huawei>
```

Step 2 After configuration is valid, then you can run the following command to build the image.

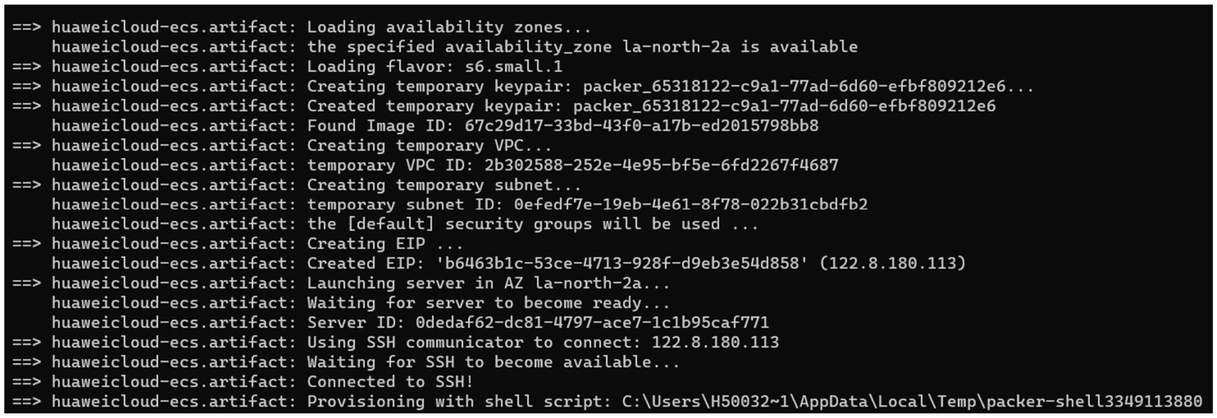
```
$ packer build hwcloud.pkr.hcl -var-file=”variables.pkrvars.hcl”
```

Using the ***.auto.pkrvats.hcl**

$ packer build .

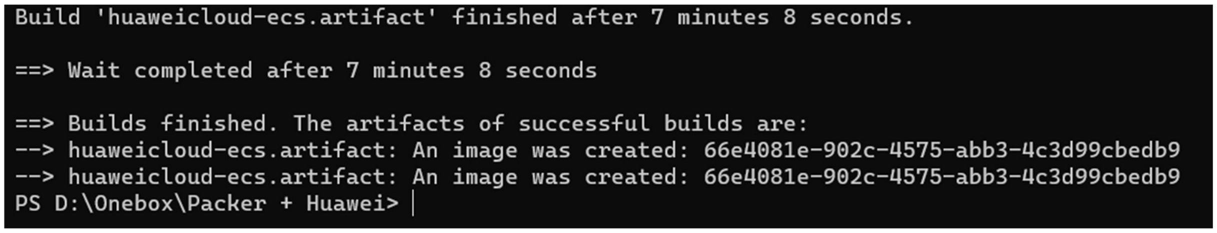
Step 3 Validate results.

A successful execution will show the following:



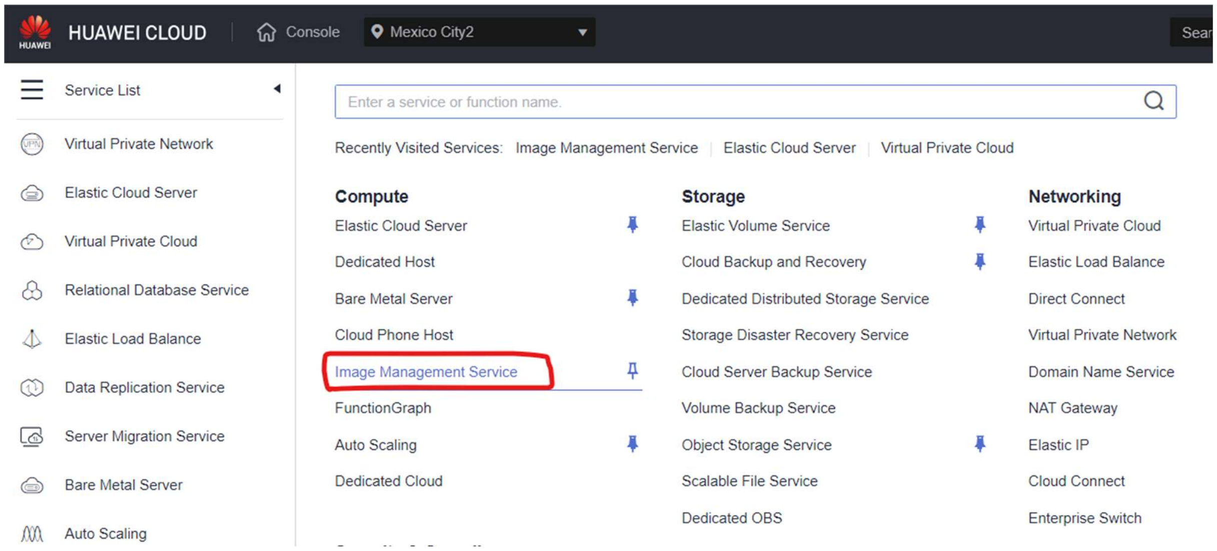
```
==> huaweicloud-ecs.artifact: Loading availability zones...
    huaweicloud-ecs.artifact: the specified availability_zone la-north-2a is available
==> huaweicloud-ecs.artifact: Loading flavor: s6.small.1
==> huaweicloud-ecs.artifact: Creating temporary keypair: packer_65318122-c9a1-77ad-6d60-efbf809212e6...
==> huaweicloud-ecs.artifact: Created temporary keypair: packer_65318122-c9a1-77ad-6d60-efbf809212e6
    huaweicloud-ecs.artifact: Found Image ID: 67c29d17-33bd-43f0-a17b-ed2015798bb8
==> huaweicloud-ecs.artifact: Creating temporary VPC...
    huaweicloud-ecs.artifact: temporary VPC ID: 2b302588-252e-4e95-bf5e-6fd2267f4687
==> huaweicloud-ecs.artifact: Creating temporary subnet...
    huaweicloud-ecs.artifact: temporary subnet ID: 0efedf7e-19eb-4e61-8f78-022b31cbdfb2
    huaweicloud-ecs.artifact: the [default] security groups will be used ...
==> huaweicloud-ecs.artifact: Creating EIP ...
    huaweicloud-ecs.artifact: Created EIP: 'b6463b1c-53ce-4713-928f-d9eb3e54d858' (122.8.180.113)
==> huaweicloud-ecs.artifact: Launching server in AZ la-north-2a...
    huaweicloud-ecs.artifact: Waiting for server to become ready...
    huaweicloud-ecs.artifact: Server ID: 0dedaf62-dc81-4797-ace7-1c1b95caf771
==> huaweicloud-ecs.artifact: Using SSH communicator to connect: 122.8.180.113
==> huaweicloud-ecs.artifact: Waiting for SSH to become available...
==> huaweicloud-ecs.artifact: Connected to SSH!
==> huaweicloud-ecs.artifact: Provisioning with shell script: C:\Users\H50032~1\AppData\Local\Temp\packer-shell3349113880
```

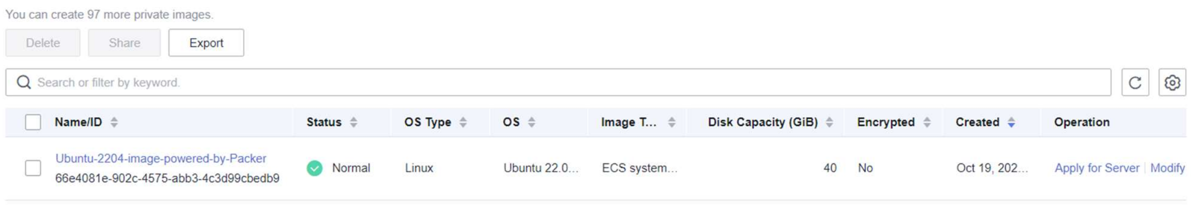
After finishing the script will inform if either success of failure happened.



```
Build 'huaweicloud-ecs.artifact' finished after 7 minutes 8 seconds.

==> Wait completed after 7 minutes 8 seconds

==> Builds finished. The artifacts of successful builds are:
--> huaweicloud-ecs.artifact: An image was created: 66e4081e-902c-4575-abb3-4c3d99cbedb9
--> huaweicloud-ecs.artifact: An image was created: 66e4081e-902c-4575-abb3-4c3d99cbedb9
PS D:\Onebox\Packer + Huawei>
```

Step 4 From the console head to the **Service List > Compute > Image Management Service**





You can create 97 more private images.

Delete Share Export

Name/ID	Status	OS Type	OS	Image T...	Disk Capacity (GiB)	Encrypted	Created	Operation
Ubuntu-2204-image-powered-by-Packer 66e4081e-902c-4575-abb3-4c3d99cbedb9	Normal	Linux	Ubuntu 22.0...	ECS system...	40	No	Oct 19, 202...	Apply for Server | Modify

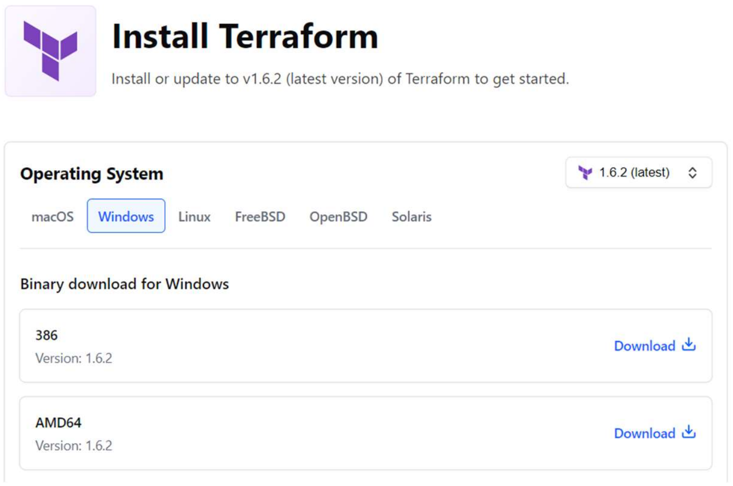
This new image can now be used to generate new ECS, through Terraform. Details on how that can be done will be described on the next section.

1.5 Using Terraform to create ECS using the generated image from Packer.

Either inside an ECS or a local machine you can download and install terraform, access the download page of the product through this link. Hashicorp Terraform Download.

As Terraform is also a multiplatform program, select the version appropriate to your OS. In this document will follow installation as if using a Windows Machine.

Step 1 Enter the download link for your appropriate platform.



Install Terraform

Install or update to v1.6.2 (latest version) of Terraform to get started.

Operating System

1.6.2 (latest)

macOS Windows Linux FreeBSD OpenBSD Solaris

Binary download for Windows

386
Version: 1.6.2
Download

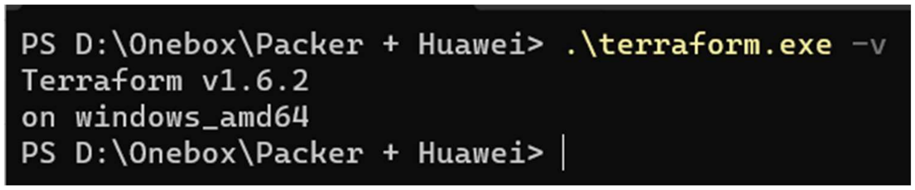
AMD64
Version: 1.6.2
Download

Step 2 Unzip the downloaded file.

Step 3 Add the Terraform package to path (Optional)

Step 4 Validate install by running the following command.

```
$ terraform -v
```



```
PS D:\Onebox\Packer + Huawei> .\terraform.exe -v
Terraform v1.6.2
on windows_amd64
PS D:\Onebox\Packer + Huawei> |
```

1.5.1.1 *Creating the Terraform Script*

Just as packer, terraform scripts are written in HCL or optionally in JSON and their extension is ".TF", It is structured by blocks. In the next steps we will generate a template for using Terraform with Huawei Cloud.

Two files need to be generated when using terraform, the versions.tf and main.tf

Step 1 Create the file versions.tf with the following information.

```
terraform {
  required_providers {
    huaweicloud = {
      source = "huaweicloud/huaweicloud"
      version = " >= 1.20.0"
    }
  }
```

Step 2 Create the main.tf file to configure the HUAWEI CLOUD provider.

```
# Defining variables for Region, and AK/SK.
variable "access_key" {
  type    = string
  default = null
}

variable "secret_key" {
  type    = string
  default = null
}

variable "region" {
  type    = string
  default = null
}

# Configure the HUAWEI CLOUD provider.
provider "huaweicloud" {
    region = var.region
    access_key  = var.access_key
    secret_key  = var.secret_key
}
```

NOTE

Remember that is **unsafe to store access and secret keys as plaintext in code**. A similar method can be used as in the packer template to create a variable file which will store the AK/SK without it being embedded in the main script.

Step 3 Assign values to the variables by creating a new file named **terraform.auto.tfvars.**

```
access_key ="TYPE YOUR AK HERE"
secret_key ="TYPE YOUR SK HERE "
region     ="SPECIFY A REGION"    # example: la-south-2
```

Just like with packer, terraform will look for any file that matches the expression ***.auto.tfvars**, avoiding the use of the -var-file flag.

Step 4 Inside the same file you can start adding resources to be created. First, add the following block to create a VPC.

```
# Create a VPC
resource "huaweicloud_vpc" "example" {
    name = "terraform_vpc"
    cidr = "192.168.0.0/16"

}
```

NOTE

You can consult all the resources that can be created by checking the Official Huawei Cloud Provider Documentation.

Step 5 Create a subnet for the ECS to attach into

```
# Creating a subnet inside the VPC
resource "huaweicloud_vpc_subnet" "subnet-example-tf" {
    name        = "subnet-terraform"
    cidr        = "192.168.1.0/24"
    gateway_ip  = "192.168.1.254"
    vpc_id      = huaweicloud_vpc.example.id
}
```

Step 6 Query the availability zones and flavors with the following data blocks.

```
data "huaweicloud_availability_zones" "myaz" {}

data "huaweicloud_compute_flavors" "myflavor" {
    availability_zone  =
data.huaweicloud_availability_zones.myaz.names[0]
    performance_type     = "normal"
    cpu_core_count       = 1
    memory_size          = 2

}
```

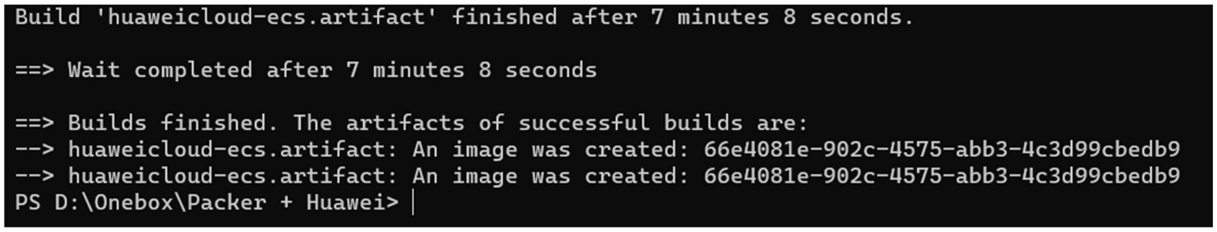
Step 7 Create a random password for the ECS. Using the **random_password** resource.

```
resource "random_password" "password" {
  length             = 16
  special            = true
  override_special   = "!@#$%*"
}
```

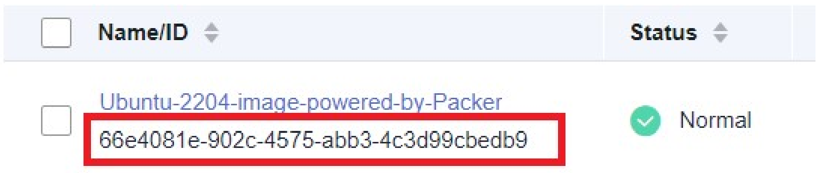
Step 8 Create the ECS using the **huaweicloud_compute_instance** resource block.

```
resource "huaweicloud_compute_instance" "basic" {
  name              = "TYPE IMAGE NAME HERE"
  admin_pass        = random_password.password.result
  image_id          = "TYPE THE IMAGE ID HERE"
  flavor_id         = data.huaweicloud_compute_flavors.myflavor.ids[0]
  availability_zone = data.huaweicloud_availability_zones.myaz.names[0]
  security_groups   = ["default"]

  network {
    uuid = huaweicloud_vpc_subnet.subnet-example-tf.id
  }
}
```

Step 9 Fill the image ID filed with the ID of the recently created image either from the packer output message, or checking from within IMS.



```
Build 'huaweicloud-ecs.artifact' finished after 7 minutes 8 seconds.

==> Wait completed after 7 minutes 8 seconds

==> Builds finished. The artifacts of successful builds are:
--> huaweicloud-ecs.artifact: An image was created: 66e4081e-902c-4575-abb3-4c3d99cbedb9
--> huaweicloud-ecs.artifact: An image was created: 66e4081e-902c-4575-abb3-4c3d99cbedb9
PS D:\Onebox\Packer + Huawei>
```



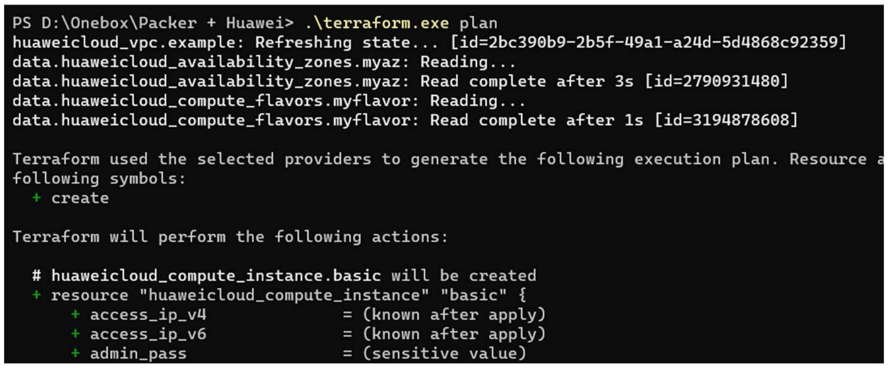
Name/ID	Status
Ubuntu-2204-image-powered-by-Packer 66e4081e-902c-4575-abb3-4c3d99cbedb9	✓ Normal

It is also possible to Terraform for querying the image ID with the data source from IMS, please refer to the Official Huawei Cloud Provider Documentation.

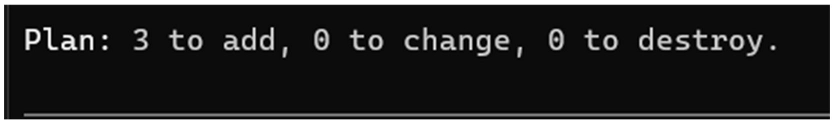
Step 10 Execute the following command to validate the resources that are going to be created.

```
$ terraform plan
```

After execution you should see this message, as only 3 resources are going to be created.



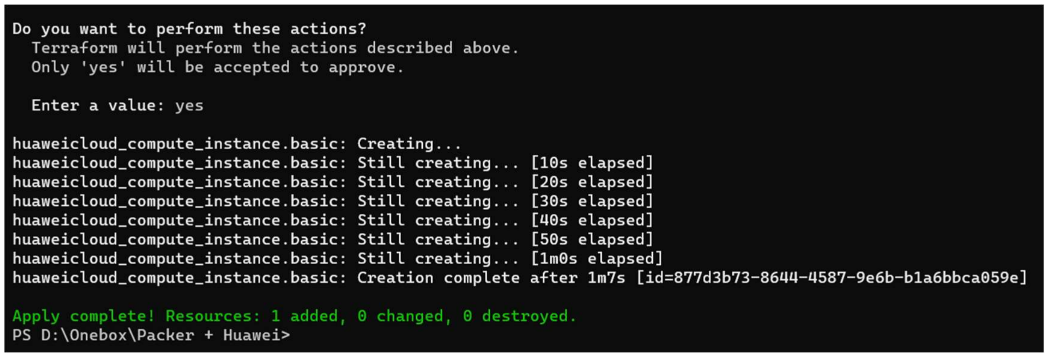
```
PS D:\Onebox\Packer + Huawei> .\terraform.exe plan
huaweicloud_vpc.example: Refreshing state... [id=2bc390b9-2b5f-49a1-a24d-5d4868c92359]
data.huaweicloud_availability_zones.myaz: Reading...
data.huaweicloud_availability_zones.myaz: Read complete after 3s [id=2790931480]
data.huaweicloud_compute_flavors.myflavor: Reading...
data.huaweicloud_compute_flavors.myflavor: Read complete after 1s [id=3194878608]

Terraform used the selected providers to generate the following execution plan. Resource a
following symbols:
  + create

Terraform will perform the following actions:

  # huaweicloud_compute_instance.basic will be created
  + resource "huaweicloud_compute_instance" "basic" {
      + access_ip_v4              = (known after apply)
      + access_ip_v6              = (known after apply)
      + admin_pass                = (sensitive value)
```



```
Plan: 3 to add, 0 to change, 0 to destroy.
```

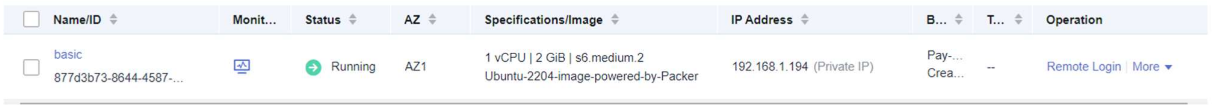
Step 11 Execute the following command to create the resources. When prompted type **"yes".**

```
$ terraform apply
```



```
Do you want to perform these actions?
  Terraform will perform the actions described above.
  Only 'yes' will be accepted to approve.

  Enter a value: yes

huaweicloud_compute_instance.basic: Creating...
huaweicloud_compute_instance.basic: Still creating... [10s elapsed]
huaweicloud_compute_instance.basic: Still creating... [20s elapsed]
huaweicloud_compute_instance.basic: Still creating... [30s elapsed]
huaweicloud_compute_instance.basic: Still creating... [40s elapsed]
huaweicloud_compute_instance.basic: Still creating... [50s elapsed]
huaweicloud_compute_instance.basic: Still creating... [1m0s elapsed]
huaweicloud_compute_instance.basic: Creation complete after 1m7s [id=877d3b73-8644-4587-9e6b-b1a6bbca059e]

Apply complete! Resources: 1 added, 0 changed, 0 destroyed.
PS D:\Onebox\Packer + Huawei>
```

You can now see your recently created ECS in the HUAWEI CLOUD console.



Name/ID	Monit...	Status	AZ	Specifications/Image	IP Address	B...	T...	Operation
basic 877d3b73-8644-4587-...		Running	AZ1	1 vCPU | 2 GiB | s6.medium.2 Ubuntu-2204-image-powered-by-Packer	192.168.1.194 (Private IP)	Pay-... Crea...	--	Remote Login | More

And as you can see it has the specifications described in the terraform script and uses image generated with packer.

2 Appendix

2.1 Source Code

2.1.1 hwcloud.pkr.hcl

```
packer {
  required_plugins {
    huaweicloud = {
      version = " >= 1.0.0"
      source  = "github.com/huaweicloud/huaweicloud"
    }
  }
}
variable "access_key" {
  type    = string
  default = null
}
variable "secret_key" {
  type    = string
  default = null
}
variable "region" {
  type    = string
  default = null
}
variable "az"{
    type    = string
    default = null
}
source "huaweicloud-ecs" "artifact" {
  access_key        = var.access_key
  secret_key        = var.secret_key
  region            = var.region
  availability_zone = var.az
  flavor            = "s6.small.1"
  source_image_name = "Ubuntu 22.04 server 64bit"
  image_name        = "Ubuntu-2204-image-powered-by-Packer"
  image_tags = {
    builder = "packer"
    os      = "Ubuntu-22.04-server"
  }
  ssh_username       = "root"
  eip_type           = "5_bgp"
  eip_bandwidth_size = 5
}
build {
  sources = ["source.huaweicloud-ecs.artifact"]

  provisioner "shell" {
    inline = ["apt-get update -y"]
  }
  post-processor "manifest" {
    strip_path = true
    output     = "packer-result.json"
  }
}
```

2.1.2 variables.auto.pkrvars

```
access_key="<AK>"
secret_key="<SK>"
region="<Region Name>"
az="<AZ Name>"
```

2.1.3 versions.tf

```
terraform {
  required_providers {
    huaweicloud = {
      source = "huaweicloud/huaweicloud"
      version = " >= 1.36.0"
    }
  }
}
```

2.1.4 terraform.auto.tfvars

```
access_key= "<AK>"
secret_key= "<SK>"
region="<Region Name>"
```

2.1.5 main.tf

```
# Defining variables for Region, and AK/SK.
variable "access_key" {
  type     = string
  default = null
}

variable "secret_key" {
  type     = string
  default = null
}

variable "region" {
  type     = string
  default = null
}

# Configure the HUAWEI CLOUD provider.
provider "huaweicloud" {
    region = var.region
    access_key  = var.access_key
    secret_key  = var.secret_key
}

# Create a VPC
resource "huaweicloud_vpc" "example" {
    name = "terraform_vpc"
    cidr = "192.168.0.0/16"

}

# Creating a subnet inside the VPC
resource "huaweicloud_vpc_subnet" "subnet-example-tf" {
    name        = "subnet-terraform"
    cidr        = "192.168.1.0/24"
    gateway_ip  = "192.168.1.254"
    vpc_id      = huaweicloud_vpc.example.id
}

data "huaweicloud_availability_zones" "myaz" {}

data "huaweicloud_compute_flavors" "myflavor" {
    availability_zone  = data.huaweicloud_availability_zones.myaz.names[0]
    performance_type     = "normal"
    cpu_core_count      = 1
    memory_size         = 2

}

resource "random_password" "password" {
  length            = 16
  special           = true
  override_special  = "!@#$%*"
}

resource "huaweicloud_compute_instance" "basic" {
  name              = "basic"
  admin_pass        = random_password.password.result
  image_id          = #TYPE YOUR IMAGE ID HERE
  flavor_id         = data.huaweicloud_compute_flavors.myflavor.ids[0]
  availability_zone = data.huaweicloud_availability_zones.myaz.names[0]
  security_group_ids   = #TYPE YOUR SG ID HERE

  network {
    uuid = huaweicloud_vpc_subnet.subnet-example-tf.id
  }

}
```

A Change History

Released On	Description
2023-10-24	First release. Hugo Juárez Pérez.