Terraform

Jem Camba - 2018-10-20 - in Orchestration

Introduction

Terraform is a tool for building, changing, and versioning infrastructure safely and efficiently, which can manage existing and popular service providers as well as custom in-house solutions.

Configuration files describe to Terraform the components needed to run a single application or your entire datacenter. Terraform generates an execution plan describing what it will do to reach the desired state, and then executes it to build the described infrastructure. As the configuration changes, Terraform is able to determine what changed and create incremental execution plans which can be applied.

The infrastructure Terraform can manage includes low-level components such as compute instances, storage, and networking, as well as high-level components such as DNS entries, SaaS features, etc.

Installing Terraform

Terraform is distributed as a binary package for all supported platforms and architectures.

To install Terraform, find the appropriate package for your system and download it. Terraform is packaged as a zip archive.

After downloading Terraform, unzip the package. Terraform runs as a single binary named terraform. Any other files in the package can be safely removed and Terraform will still function.

The final step is to make sure that the terraform binary is available on the PATH. See this page for instructions on setting the PATH on Linux and Mac. This page contains instructions for setting the PATH on Windows.

Packet & Terraform

Packet is one of the Cloud Providers that integrate with Terraform. In Terraform terms, a provider is responsible for understanding API interactions and exposing resources. Therefore, the Packet provider is used to interact with the resources supported by Packet.

You can always find the latest resources and documentation here.

Creating your first Project and Server with Terraform

Terraform uses text files to describe infrastructure and to set variables. These text files are called Terraform configurations and end in .tf.

When invoking any command that loads the Terraform configuration, Terraform loads all configuration files within the directory specified in alphabetical order.

In order to use Terraform with Packet, you will need to have an API token ready.

Here's how your first configuration file will look:

```
user@master$ cat sample_terraform.tf

# Configure the Packet Provider

provider "packet" {

 auth_token = "{API_TOKEN}"

}

# Create a project

resource "packet_project" "simple_project" {

 name          = "Terraform Project"

}
```

In this example, we will be able to create a new project, named Terraform Project.

Initialize the packet provider by running `terraform init`

```
user@master$ terraform init

Initializing provider plugins...
```

- Checking for available provider plugins on https://releases.hashicorp.com...

- Downloading plugin for provider "packet" (1.2.0)...

...

...

* provider.packet: version = "~> 1.2"

Terraform has been successfully initialized!

To create your first project run `terraform apply`

user@master$ terraform apply

An execution plan has been generated and is shown below.

Resource actions are indicated with the following symbols:

+ create

Terraform will perform the following actions:

+ packet_project.simple_project

id: <computed>

created: <computed>

name: "Terraform Project"

updated: <computed>

Plan: 1 to add, 0 to change, 0 to destroy.

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

packet_project.simple_project: Creating...

  created: "" => "<computed>"

  name:    "" => "Terraform Project"

  updated: "" => "<computed>"

packet_project.simple_project: Creation complete after 1s (ID: 667746a1-
d44b-44a5-9784-becc437aab87)

Apply complete! Resources: 1 added, 0 changed, 0 destroyed.
```

To deploy a server we can add the following code at the bottom of our sample_terraform.tf configuration file and run `terraform apply` again.

```
resource "packet_device" "host1" {

  hostname         = "tf.host1"

  plan             = "baremetal_0"

  facility         = "ewr1"

  operating_system = "ubuntu_16_04"

  billing_cycle    = "hourly"

  project_id       = "${packet_project.simple_project.id}"

}

user@master$ terraform apply

packet_project.simple_project: Refreshing state... (ID: 667746a1-d44b-44a5-9784-
becc437aab87)

An execution plan has been generated and is shown below.

Resource actions are indicated with the following symbols:
```

```
  + create

Terraform will perform the following actions:

  + packet_device.host1

      id:                     <computed>

      always_pxe:             "false"

      billing_cycle:          "hourly"

      created:                <computed>

      facility:               "ewr1"

      hardware_reservation_id: <computed>

      hostname:               "tf.host1"

      operating_system:       "ubuntu_16_04"

      plan:                   "baremetal_0"

      project_id:             "667746a1-d44b-44a5-9784-becc437aab87"

Plan: 1 to add, 0 to change, 0 to destroy.

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

packet_device.host1: Creating...

  always_pxe:             "" => "false"

  billing_cycle:          "" => "hourly"
```

```
  created:                "" => "<computed>"
  facility:               "" => "ewr1"
  hardware_reservation_id: "" => "<computed>"
  hostname:               "" => "tf.host1"
  locked:                 "" => "<computed>"
  network.#:              "" => "<computed>"
  operating_system:       "" => "ubuntu_16_04"
  plan:                   "" => "baremetal_0"
  project_id:             "" => "667746a1-d44b-44a5-9784-becc437aab87"
  public_ipv4_subnet_size: "" => "<computed>"
  root_password:          "<sensitive>" => "<sensitive>"
  state:                  "" => "<computed>"
  updated:                "" => "<computed>"
packet_device.host1: Still creating... (10s elapsed)
packet_device.host1: Still creating... (20s elapsed)
packet_device.host1: Still creating... (30s elapsed)
…
…
packet_device.host1: Still creating... (4m40s elapsed)
packet_device.host1: Still creating... (4m50s elapsed)
packet_device.host1: Still creating... (5m0s elapsed)
```

```
packet_device.host1: Creation complete after 5m2s (ID: 35f76adc-b6e7-4ac4-8160-bddea2b80d74)

Apply complete! Resources: 1 added, 0 changed, 0 destroyed.
```

To get additional information about the server, including the management IPs, UUID, or the root password, you can run the `terraform show` command.

```
user@master$ terraform show

packet_device.host1:

  id = 35f76adc-b6e7-4ac4-8160-bddea2b80d74

  created = 2018-03-30T01:19:51Z

  facility = ewr1

  hostname = tf.host1

  network.0.address =

  operating_system = ubuntu_16_04

  plan = baremetal_0

  project_id = 667746a1-d44b-44a5-9784-becc437aab87

  public_ipv4_subnet_size = 31

  root_password =

  state = active

  tags.# = 0

  updated = 2018-03-30T01:24:51Z

packet_project.simple_project:

  id = 667746a1-d44b-44a5-9784-becc437aab87
```

```
  created = 2018-03-30T01:18:49Z

  name = Terraform Project

  updated = 2018-03-30T01:18:49Z
```

To delete a device remove the server declaration resource from the sample_terraform.tf configuration file, and run `terraform apply` again.

```
user@master$ terraform apply

packet_project.simple_project: Refreshing state... (ID: 667746a1-d44b-44a5-9784-becc437aab87)

packet_device.host1: Refreshing state... (ID: 35f76adc-b6e7-4ac4-8160-bddea2b80d74)

An execution plan has been generated and is shown below.

Resource actions are indicated with the following symbols:

  - destroy

Terraform will perform the following actions:

  - packet_device.host1

Plan: 0 to add, 0 to change, 1 to destroy.

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

packet_device.host1: Destroying... (ID: 35f76adc-b6e7-4ac4-8160-bddea2b80d74)

packet_device.host1: Destruction complete after 0s

Apply complete! Resources: 0 added, 0 changed, 1 destroyed.
```

Deploy a server with Block Storage

This is how the sample_terraform.tf would look:

```
# Configure the Packet Provider

provider "packet" {

 auth_token = "{API_TOKEN}"

}

# Create a project

resource "packet_project" "simple_project" {

 name           = "Terraform Project"

}

# Create a device

resource "packet_device" "host1" {

    hostname         = "tf.node1"

    plan             = "baremetal_0"

    facility         = "ewr1"

    operating_system = "ubuntu_16_04"

    billing_cycle    = "hourly"

    project_id       = "${packet_project.simple_project.id}"

    user_data        = "${file("volume-attach-userdata")}"

 }

# Create a volume

resource "packet_volume" "volume1" {
```

```
    plan = "storage_2"
    billing_cycle = "hourly"
    size = 250
    project_id = "${packet_project.simple_project.id}"
    facility = "ewr1"
  }
#Attach volume to the server
resource "packet_volume_attachment" "volume1_attachment" {
    device_id = "${packet_device.host1.id}"
    volume_id = "${packet_volume.volume1.id}"
  }
```

The script that will attach the volume within the OS using userdata

```
#!/bin/bash
apt-get update -y
VLM=$(curl -s https://metadata.packet.net/metadata | jq -r '.volumes[] | .name')
sleep 5
packet-block-storage-attach -m queue -v
echo -e "o\nn\np\n1\n\n\nw" | fdisk /dev/mapper/$VLM
sleep 5
kpartx -u /dev/mapper/$VLM-part1
mkfs.ext4 /dev/mapper/$VLM-part1
```

```
mkdir /mnt/block

mount -t ext4 /dev/mapper/$VLM-part1 /mnt/block

echo "/dev/mapper/$VLM-part1 /mnt/block ext4 _netdev 0 0" >> /etc/fstab

echo "Hello World" >> /mnt/block/file
```

Once both files are you to your liking run `terraform apply`

```
packet_project.simple_project: Refreshing state... (ID: 667746a1-d44b-44a5-9784-becc437aab87)

An execution plan has been generated and is shown below.

Resource actions are indicated with the following symbols:

  + create

Terraform will perform the following actions:

  + packet_device.host1

      id:                   <computed>

      always_pxe:           "false"

      billing_cycle:        "hourly"

      created:              <computed>

      facility:             "ewr1"

      hostname:             "tf.node1"

      operating_system:     "ubuntu_16_04"

      plan:                 "baremetal_0"

      project_id:           "667746a1-d44b-44a5-9784-becc437aab87"
```

```
  + packet_volume.volume1
      id:                   <computed>
      facility:             "ewr1"
      name:                 <computed>
      plan:                 "storage_2"
      project_id:           "667746a1-d44b-44a5-9784-becc437aab87"
      size:                 "250"


  + packet_volume_attachment.volume1_attachment
      id:                   <computed>
      device_id:            "${packet_device.host1.id}"
      volume_id:            "${packet_volume.volume1.id}"
Plan: 3 to add, 0 to change, 0 to destroy.
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
packet_volume.volume1: Creating...
  attachments.#: "" => "<computed>"
  billing_cycle: "" => "hourly"
  created:       "" => "<computed>"
```

```
  facility:       "" => "ewr1"
  name:           "" => "<computed>"
  plan:           "" => "storage_2"
  project_id:     "" => "667746a1-d44b-44a5-9784-becc437aab87"
  size:           "" => "250"
  state:          "" => "<computed>"
  updated:        "" => "<computed>"
packet_device.host1: Creating...
  always_pxe:               "" => "false"
  billing_cycle:            "" => "hourly"
  created:                  "" => "<computed>"
  facility:                 "" => "ewr1"
  hostname:                 "" => "tf.node1"
  operating_system:         "" => "ubuntu_16_04"
  plan:                     "" => "baremetal_0"
  project_id:               "" => "667746a1-d44b-44a5-9784-becc437aab87"
packet_volume.volume1: Still creating... (10s elapsed)
packet_device.host1: Still creating... (10s elapsed)
packet_volume.volume1: Creation complete after 14s (ID: 09adf5ad-
e728-47ef-9428-78f253fb6303)
packet_device.host1: Still creating... (20s elapsed)
```

```
...

...

packet_device.host1: Still creating... (4m10s elapsed)

packet_device.host1: Still creating... (4m20s elapsed)

packet_device.host1: Creation complete after 4m29s (ID: 6ce691c9-aeaa-471d-
a39a-6e84416bf447)

packet_volume_attachment.volume1_attachment: Creating...

  device_id: "" => "6ce691c9-aeaa-471d-a39a-6e84416bf447"

  volume_id: "" => "09adf5ad-e728-47ef-9428-78f253fb6303"

packet_volume_attachment.volume1_attachment: Creation complete after 1s (ID:
94ac72a4-1bd4-4039-9420-394ba0a0c0be)

Apply complete! Resources: 3 added, 0 changed, 0 destroyed.
```

To ensure things went off without a hitch, check to see if your hello world is found:

```
ssh -t root@deviceip "cat /mnt/block/file"
```

Tags
terraform