LAB-03-Report-ist189399.md

Webfront Lab README

AGISIT 20201-2022

Authors

Team 20A

Number	Name	Username	Email
ist189399	Afonso Goncalves	https://git.rnl.tecnico.ulisboa.pt/ist189399	mailto:afonso.corte-real.goncalves@tecnico.ulisboa.pt
ist190621	Maria Filipe	https://git.rnl.tecnico.ulisboa.pt/ist190621	mailto:maria.j.d.c.filipe@tecnico.ulisboa.pt
ist189498	Maria Martins	https://git.rnl.tecnico.ulisboa.pt/ist189498	mailto:maria.d.martins@tecnico.ulisboa.pt

Q01

When you run the command `terraform init` which plugins were installed? (You can copy the result of the command to insert in your report).

```
vagrant@mgmt:~/labs/gcpcloud$ terraform init

Initializing the backend...

Initializing provider plugins...
- Finding latest version of hashicorp/google...
- Installing hashicorp/google v3.89.0...
- Installed hashicorp/google v3.89.0 (signed by HashiCorp)

Terraform has created a lock file .terraform.lock.hcl to record the provider
selections it made above. Include this file in your version control repository
so that Terraform can guarantee to make the same selections by default when
you run "terraform init" in the future.

Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform plan" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.

If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
vagrant@mgmt:~/labs/gcpcloud$
```

Q02

Analyze briefly the `terraform-gcp-servers.tf` and interpret its purpose.

Line no	Interpretation

Line no	Interpretation
10	Creates 3 instances
11, 38	Sets name for each inscance
12, 39	Sets machine type as `n1-standard-1` (according to variable)
13, 40	Sets region to `europe-west4` (according to variable)
15-21, 42-48	Sets `ubuntu-2004-focal-v20210927 (according to variable) as boot image
23-27, 50-54	Attaches network interfact to default network
25-26, 52-53	Assigns an external IP to the machine
29-31, 56-58	Attaches mgmt public key to VM instance (allows passwordless ssh)
32, 60	Adds network tag to machine

Q03

> Analyze briefly the `terraform-gcp-variables.tf` and interpret its purpose.

This file declares configuration variables that will be used in other terraform files. This way, if something needs to be changed, it is only changed once!

Line no	Interpretation
6-8	Sets a variable for the project name
15-17	Sets a variable for the machine type
23-25	Sets a variable for the region the vms will run
28-30	Sets a variable for the disk size

Q04

> If you would need more Web servers where and how would you declare that intention (with Terraform)?

We could change the `count` property in file `terraform-gcp-servers.tf:10` to the desired number of web servers.

Q05

> Which other files appeared in the `gcpcloud` folder after running `terraform apply` for your infrastructure?

After running `terraform apply`, we could see that terraform created the files `terraform.tfstate` and `terraform.tfstate.backup`. The first file contains the state information of the remote cloud infratstructure terraform just created (can be updated with `terraform refresh`). The latest file is a backup in case the former is somehow lost or currupted.

Q06

> After creating the infrastructure with Terraform, you needed to modify the `gcphosts` inventory. What was changed in that file, and what was the purpose?

We used the output of the `terraform apply` command to gather the IPs of the created machines. We then added those IPs to the ansible inventory (in each line, we set the IP corresponding to the hostname), so that ansible knew where to connect when running the plays.

Q07

> After creating the infrastructure with Terraform, you have tested the communication with the remote instances using an Ansible ad-hoc command. Copy the result into your report. Was the result successful? In case of positive answer, can you explain why it worked, i.e., how was Ansible capable of establishing a session with the remote instances?

The connectivity was successful:

```
vagrant@mgmt:~/labs/gcpcloud$ ansible targets -m ping
balancer | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
web1 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
web3 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
web2 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
```

The `ssh` connection succeeded since every VM already contained the `mgmt` public key in their `~/.ssh/authorized_hosts` file. This key was installed when the machines were created. This occured since lines `31` and `44` of `terraform-vmcloud-servers.tf` attach this same public key to these machines.

Q08

> Which files did you had to modify, so that Ansible could Configure correctly the instances? Explain in detail what were the modifications.

Ansible couldn't generate `haproxy.cfg` nor `index.html` from templates since in the playbook, the `src` value contained an invalid path. After fixing this error, ansible run correctly with no trouble.

Q09

> When the system was fully deployed, when hitting the refresh button on the web browser with the address of the Load Balancer, What changed? Describe those changes and interpret why they happened (if indeed happened). Which IP addresses of the instances were Shown, and what do they correspond to?

At each refresh, the presented page would change according to the web server that was serving it. This happened since the balancer was alternating the servers in a roundrobin mode, as configured in the `haproxy.cfg.j2:54` template.

Q10

> When you run `terraform destroy` what was the result after you have confirmed the question "Do you really want to destroy all resources?". Copy those results to your report. After the destroy, looking at the Google Cloud Platform Dashboard, namely in the ACTIVITY tab, what information is displayed so that you are sure the resources have really been destroyed?

After running `terraform destroy` command, we could see the following output:

```
o you really want to destroy all resources?
 Terraform will destroy all your managed infrastructure, as shown above.
 There is no undo. Only 'yes' will be accepted to confirm.

 Enter a value:
oogle_compute_instance.web[2]: Destroying... [id=projects/agisit-2021-kube-team20a/zones/europe-west4-a/instances/web3]
oogle_compute_instance.web[0]: Destroying... [id=projects/agisit-2021-kube-team20a/zones/europe-west4-a/instances/web1]
oogle_compute_firewall.frontend_rules: Destroying... [id=projects/agisit-2021-kube-team20a/global/firewalls/frontend]
oogle_compute_instance.balancer: Destroying... [id=projects/agisit-2021-kube-team20a/zones/europe-west4-a/instances/balancer]
oogle_compute_instance.web[1]: Destroying... [id=projects/agisit-2021-kube-team20a/zones/europe-west4-a/instances/web2]
oogle_compute_instance.web[2]: Still destroying... [id=projects/agisit-2021-kube-team20a/zones/europe-west4-a/instances/web3, 10s elapsed]
oogle_compute_firewall.frontend_rules: Still destroying... [id=projects/agisit-2021-kube-team20a/global/firewalls/frontend, 10s elapsed]
oogle_compute_instance.web[0]: Still destroying... [id=projects/agisit-2021-kube-team20a/zones/europe-west4-a/instances/web1, 10s elapsed]
oogle_compute_instance.web[1]: Still destroying... [id=projects/agisit-2021-kube-team20a/zones/europe-west4-a/instances/web2, 10s elapsed]
oogle_compute_instance.balancer: Still destroying... [id=projects/agisit-2021-kube-team20a/zones/europe-west4-a/instances/balancer, 10s elapsed]
oogle_compute_firewall.frontend_rules: Destruction complete after 11s
oogle_compute_instance.web[2]: Still destroying... [id=projects/agisit-2021-kube-team20a/zones/europe-west4-a/instances/web3, 20s elapsed]
oogle_compute_instance.web[0]: Still destroying... [id=projects/agisit-2021-kube-team20a/zones/europe-west4-a/instances/web1, 20s elapsed]
oogle_compute_instance.balancer: Still destroying... [id=projects/agisit-2021-kube-team20a/zones/europe-west4-a/instances/balancer, 20s elapsed]
oogle_compute_instance.web[1]: Still destroying... [id=projects/agisit-2021-kube-team20a/zones/europe-west4-a/instances/web2, 20s elapsed]
oogle_compute_instance.web[1]: Destruction complete after 22s
oogle_compute_instance.balancer: Destruction complete after 22s
oogle_compute_instance.web[0]: Destruction complete after 22s
oogle_compute_instance.web[2]: Destruction complete after 22s

estroy complete! Resources: 5 destroyed.
agrant@mgmt:~/labs/gcpcloud$
```

Meaning that the virtual machines were successfully destroyed. When checking the GCP Dashboard, we could confirm that the operation was successful: