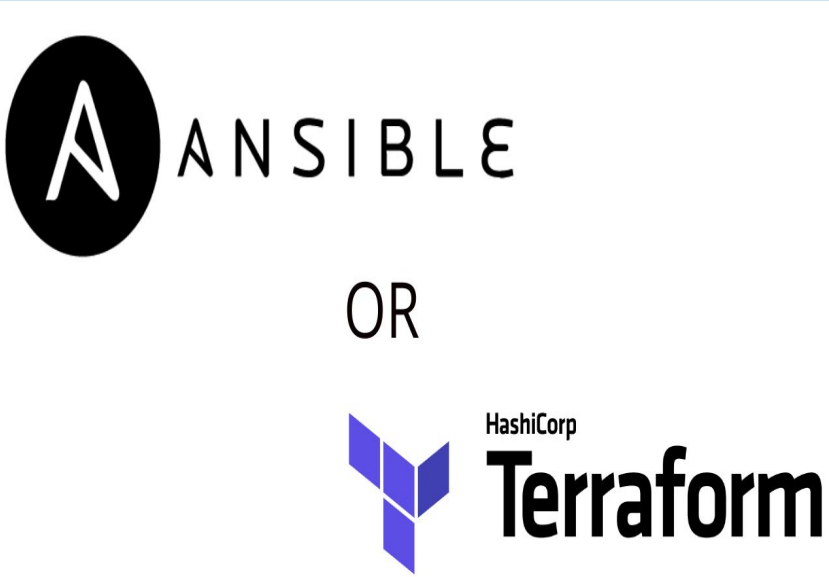
Infrastructure as Code (with Terraform)

Terraform vs. Ansible

12.02.2021

Gökhan Yildirim, Samir Hamiani, Victoria Chaikovska

Frankfurt University of Applied Sciences
Faculty of Computer Science and Engineering



Outline

- Infrastructure as a Code
- Infrastructure as a code (Terraform)
- Terraform lifecycle
- Infrastructure as a code (Ansible)
- Ansible playbook
- Introduction of the Infrastructure
- Terraform and Ansible Live Demo
- Comparison Ansible and Terraform
- Summary

Infrastructure as a code

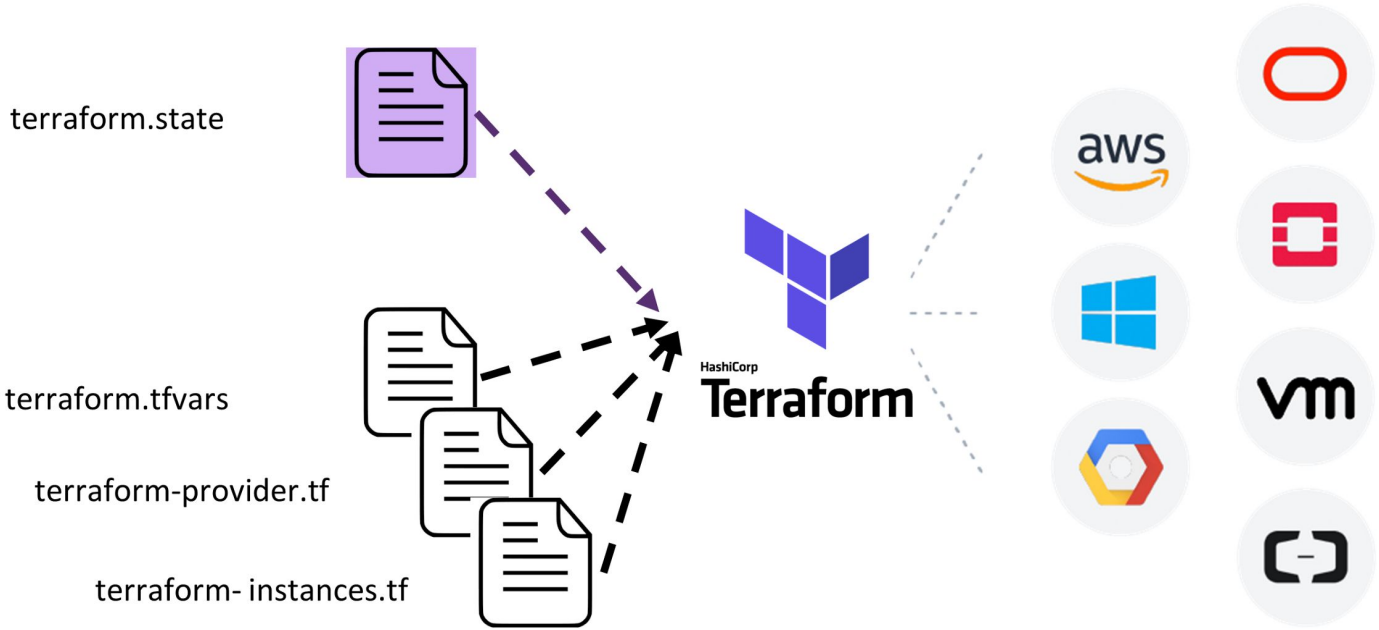
- “Infrastructure as Code (IaC) is the managing and provisioning of infrastructure through code instead of through manual processes.” [1]
- There are 2 types of IaC tools :
  - configuration management tools
  - orchestration
- Advantages of Iac:
  - simplicity
  - efficiency and speed
  - low risk
  - Costs

[1] What is Infrastructure as Code (IaC)?URL: https://www.redhat.com/en/what-is-infrastructure-as-code-iac

Infrastructure as a code (Terraform)

- Engine which allows to develop and modify infrastructures
- On various types of providers
- Simple syntax allows simple modularity and works well with multi-cloud systems
- Managing IaC is also a foundation for DevOps practices
- The main language of terraform is HCL

Infrastructure as a code (Terraform)



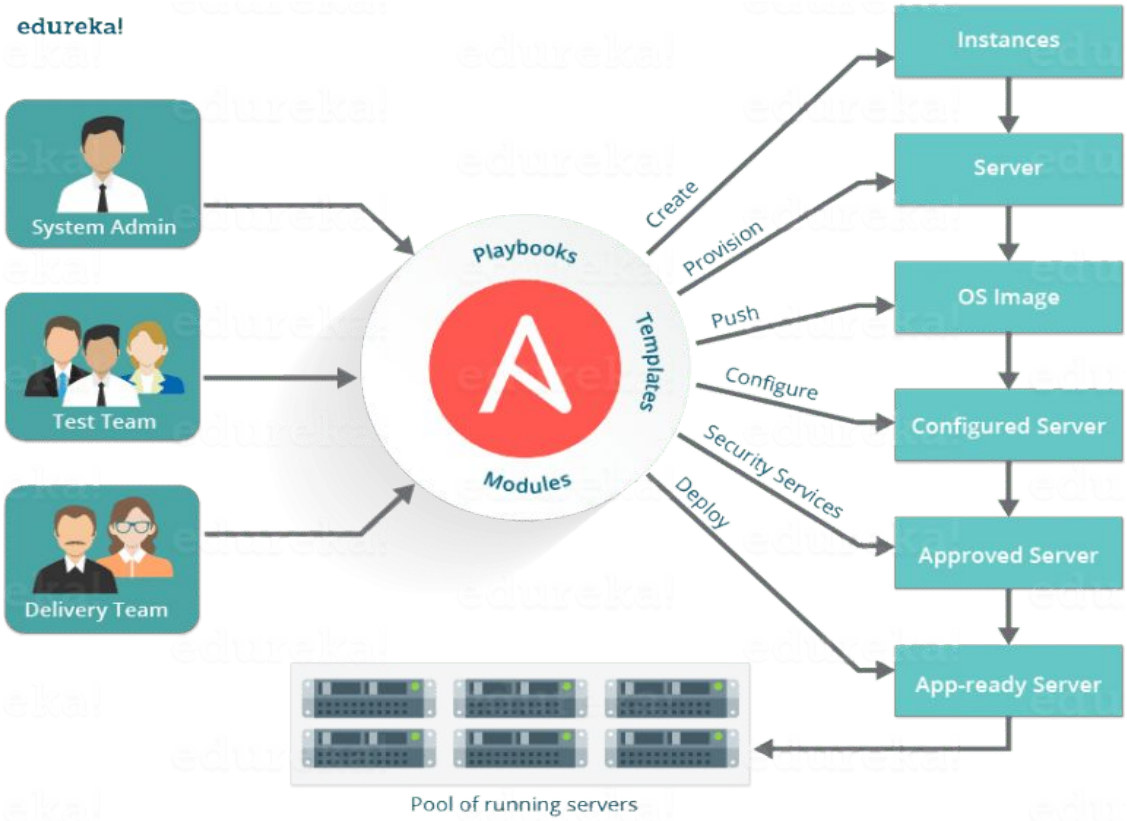
Terraform lifecycle

- **terraform init -** initialize the working directory in
- **terraform validate -** to check the written code for syntax errors
- **terraform refresh -** to coordinate the actual state
- ***terraform plan* -** for creating an execution plan
- ***terraform apply* -** the creation of our infrastructure.
- ***terraform destroy* -** destroying the infrastructure

Infrastructure as a code (Ansible)

- Open-Source tool for providing infrastructure as code
- Ansible configure slave nodes
- Configurations of the slaves are done with Ansible modules
- Modules are written in YAML and include a routine of tasks
- Modules can be executed in the console or in Playbook
- Playbooks describes the commands to achieve the desired state
- This state can be basic settings or a complete setup

Infrastructure as a code (Ansible)



Ansible playbook

- The list of all configs existing in the control node, command `$ ansible-config`
- Specifying User:

```
$ ansible-playbook FileName.yml --user muser
```

- Run Ansible:

```
$ ansible-playbook FileName.yml
```

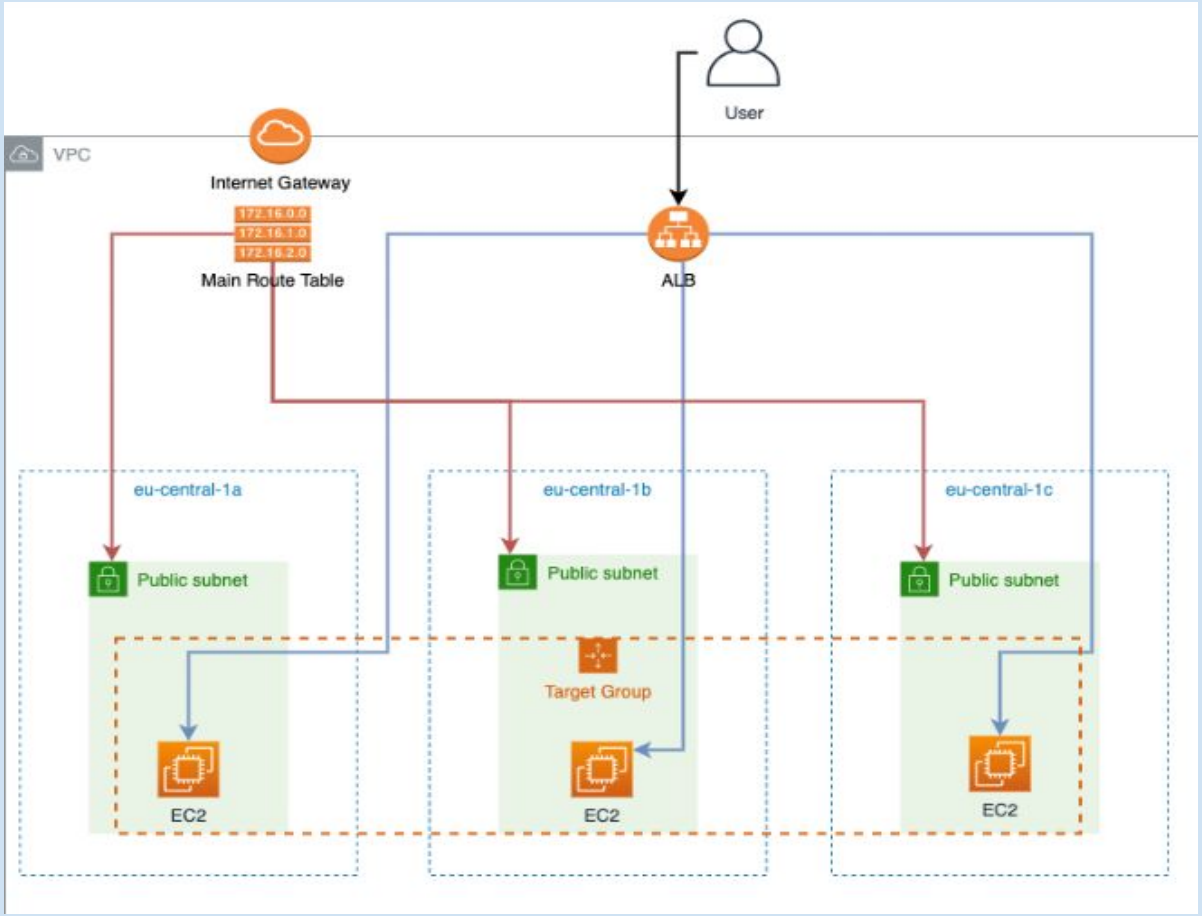
- Check bad syntax:

```
$ ansible-playbook NAME.yml --syntax-check
```

- Running a playbook in dry-run mode:

```
$ ansible-playbook playbooks/PLAYBOOK_NAME.yml --check
```

Introduction of the Infrastructure



Terraform and Ansible
LIVE DEMO

Comparison Terraform and Ansible

Terraform

- **Type:** Mainly infrastructure Orchestration tool
- **Support:** Only partial Support for packaging and templating. Terraform offers direct access to HashiCorp's support
- **Ease set-up and usage:** Tool is simpler to use and to set-up
- **Lifecycle management:** Lifecycle management

Ansible

- **Type:** Mainly configuration tool Install/Update software on that infrastructure
- **Support:** Complete Support for packaging and templating.
- **Ease set-up and usage:** It is easy to install and use. The tool has a master without agents (agentless)
- **Lifecycle management:** No Lifecycle management

Comparison Terraform and Ansible

Terraform

- **Infrastructure:** Provides support for immutable infrastructure
- **Availability:** Not Applicable
- **Modules:** The modules offer for users an abstract away of any reusable parts.
- **GUI:** Only 1/3 parts of GUIs are available
- **Language:** Uses declarative language
- **Market:** Relatively new

Ansible

- **Infrastructure:** Provides support for mutable infrastructure
- **Availability:** The tool has a secondary node in case an active node not function
- **Modules:** Ansible Galaxy available, it consists of a repository or library
- **GUI:** GUI is presented as a command-line tool
- **Language:** Uses procedural language
- **Market:** More mature

Summary

- Which tool to choose? What tool is better?
- Both tools are well-known for their unambiguous advantages in creating infrastructure as a code
- These tools are very helpful in deploying repeatable environments with complex requirements
- Terraform and Ansible are automated: configuring, provisioning and managing the infrastructure

Summary

- It is recommendable to use Terraform for orchestration and Ansible configuration management
- In comparison to Terraform Ansible is more tricky in use
- Ansible takes much more time for learning, because the documentation of Ansible has only minimal basic information
- To get experience in Ansible you should start to learn automating deployments, configuration and management of the infrastructure

References

- What is Infrastructure as Code (IaC)?URL: https://www.redhat.com/en/what-is-infrastructure-as-code-iac
- Ansible vs Terraform: Understanding the Differences. URL: https://www.whizlabs.com/blog/ansible-vs-terraform/
- Ansible vs Terraform vs Puppet:Which to Choose? URL: https://phoenixnap.com/blog/ansible-vs-terraform-vs-puppet
- Running AdHoc Commands. URL: https://ansible-tips-and-tricks.readthedocs.io/en/latest/ansible/commands/

References

- DevOps101 — First Steps on Terraform: Terraform + OpenStack + Ansible. URL: https://medium.com/hackernoon/terraform-openstack-ansible-d680ea466e22
- What Is Ansible? – Configuration Management And Automation With Ansible. URL: https://www.edureka.co/blog/what-is-ansible/


ΕΥΧΑΡΙΣΤΩ
TÄNAN
HVALA
GRACIAS
DZIĘKUJĘ
TACK
GRAZIE
ありがとう
MERCI
THANK YOU
DIAKUIU
PALDIES
ACIU
TAK
DANKE
DANK U WEL
ДЗЯКУЮ
СПАСИБО
谢谢
OBRIGADO
diolch
KIITOS
TESEKKUR EDERIM