BATCH: 149

26/09/2023

DT/NT : DT

LESSON : DevOps

SUBJECT: Terraform 1

TECHPRO EDUCATION

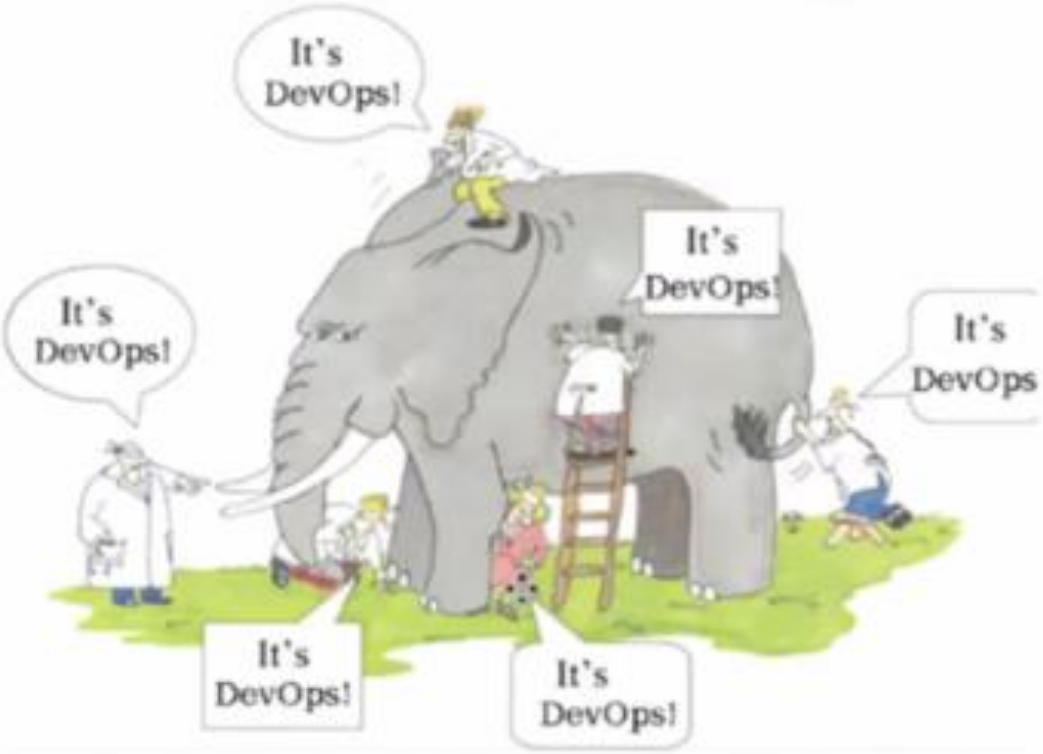
techproeducation.com

+1 (585) 304 29 59

What is DevOps

What DevOps is Not...

- a tool
- a role
- a team
- something that can be purchased or simply switched on



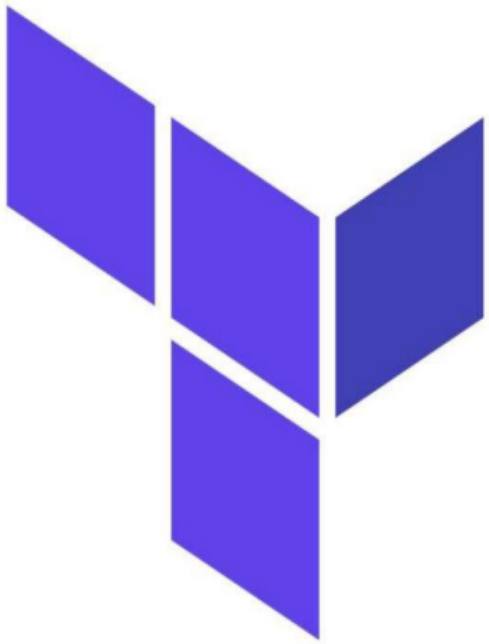
Bir çok tanımı olabilir DevOpsun.
Developer ve operatör ekiplerini birleştiren yeni bir kültürdür.

Yazılımcı güncelleme getiriyor test etmem gerekiyor diyor, Operations (eski zamanda) server ları yürütmeye göre ayarladık yeni server lazım olacak iş çıkarıyorsunuz diyor.

Ya da developer kodu yazar, çok güzel bir güncellemedir Operations a gönderir, ancak güncelleme müthiş şekilde RAM ve CPU yiyor. Bu da Operations içinde ciddi bir sorundur.

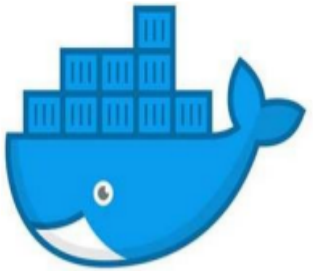
DevOps uygulamaları genellikle çeşitli araçları içerir. Bu araçlar, kod depolama, sürüm kontrolü, yapı (build) otomatizasyonu, konfigürasyon yönetimi, izleme ve loglama gibi farklı amaçlar için kullanılır. Örneğin:

- **Jenkins** - Sürekli entegrasyon ve sürekli dağıtım (CI/CD).
- **Docker** - Konteynerizasyon.
- **Kubernetes** - Konteyner orkestrasyonu.
- **Ansible/Chef/Puppet** - Konfigürasyon yönetimi.
- **Git** - Sürüm kontrolü.
- **Prometheus/Grafana** - İzleme ve metrik.
- **ELK Stack/Elastic Stack** - Log yönetimi.



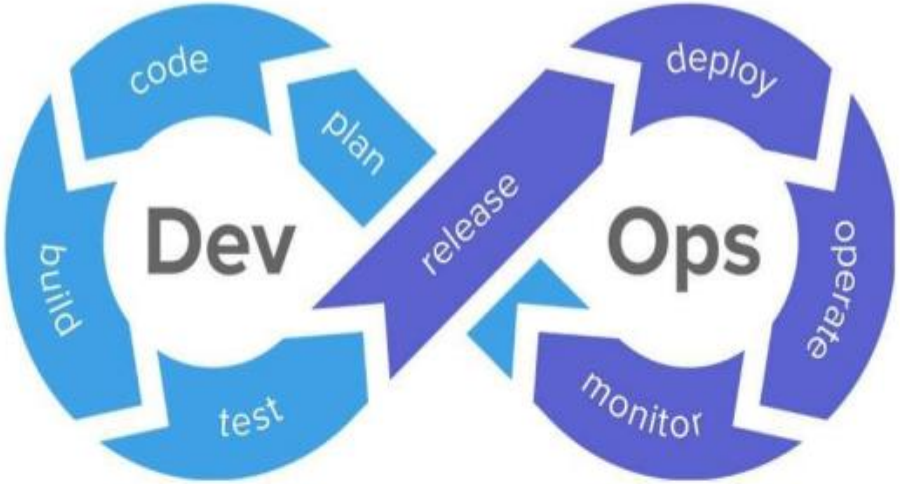
* Terraform'a giriş
* Providers
* Resources & Data Sources
* Terraform State
* Output
* Variables
* Enviroment Variables
* Modules

- Terraform'u
  - VPC(Amazon Virtual Private Cloud) oluştururken,
  - AWS kullanıcısı oluştururken ve kullanıcıyla bağlantılı izinleri yönetirken
  - Sunucuları spin ederken(Spinning Up the Servers)
  - ve Docker I yüklerken kullanırız.
  - Terraform la ayrıca yukarıdaki görevlerin doğru bir sırada yapıldığına emin oluruz.



- - Infastructure ını
- - Platform unu
- - Platformunda çalıştırdığın servisleri(AWS, Jenkins gibi) yönetmene izin verir.
- Açık kaynak kodludur.
- “Declaretive” dir. Yani her bir adımı teker teker manuel olarak kodlamazsın da ideal durumu verirsin. Terraform varolan kodu ideal duruma uyarlar.

- Provisioning infostructure(DevOps)



- Application Deploying(Developer Takımı)

Terraform Elements

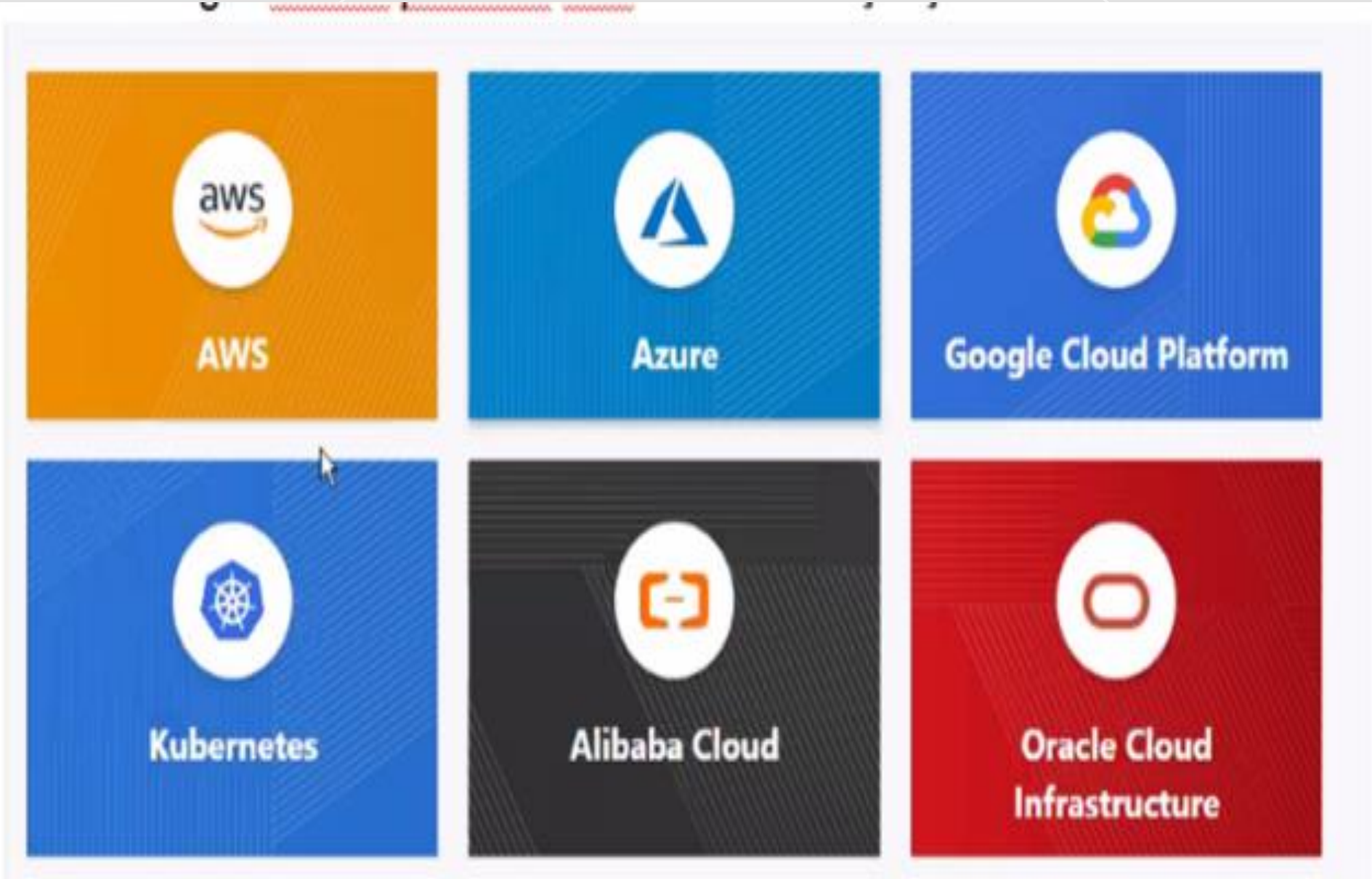
Providers

A provider is responsible for understanding API interactions and exposing resources. Every Terraform provider has its own documentation, describing its resource types and their arguments.

Bu kod ile kullanacağımız platform (burada aws) un versionunu belirtmiş oluyoruz. Provider ne demiştik, Terraform un platformlar ın API ile iletişime geçen bileşenidir.

terraform-lesson > terraform-aws > main.tf > ...

```
1   terraform {
2     required_providers {
3       aws = {
4         source = "hashicorp/aws"
5         version = "4.57.1"
6       }
7     }
8   }
9
10  provider "aws" {
11    # Configuration options
12  }
13
14
```



Terraform Elements

Modules

A Terraform module is a set of Terraform configuration files in a single directory. Even a simple configuration consisting of a single directory with one or more «.tf» files is a module. When you run Terraform commands directly from such a directory, it is considered the root module. So in this sense, every Terraform configuration is part of a module.

```
$ tree minimal-module/
.
├── LICENSE
├── README.md
├── main.tf
├── variables.tf
└── outputs.tf
```

Backends

Terraform Elements

Backends

A "backend" in Terraform determines how state is loaded and how an operation such as <apply> is executed. By default, Terraform uses the "local" backend, which is the normal behavior of Terraform you're used to. Backends are completely optional. You can successfully use Terraform without ever having to learn or use backends. Backends are used for keeping sensitive information off disk.

Agnostic

Advantages of Terraform

Platform Agnostic

- In a modern datacenter, you may have several different clouds and platforms to support your various applications.
- With Terraform, you can manage a **heterogeneous environment** with the same workflow by creating a configuration file to fit the needs of your project or organization.

Documentation

Learn More Terraform

- Terraform Documentation
- Hashicorp/terraform (Github Page)
- Shuaibiyy/awesome-terraform
- tfutils/tfenv
- gruntwork-io/terragrunt
- 28mm/blast-Radius
- Terraform Registry

Terraform & Ansible

- **FARKLAR**

Terraform

- - Genellikle Infrastructure provisioning tool(Altyapı sağlama aracı) olarak kullanılır.
- - Görece daha yenidir.
- - Orchestration yeteneği daha gelişmiştir. Orchestration, bilgisayar sistemlerinin, uygulamaların ve hizmetlerin otomatik yapılandırması, yönetimi ve koordinasyonudur. IT departmanının karmaşık görevleri ve iş akışlarını daha kolay yönetmesine yardımcı olur.
-

Ansible

- - Genellikle configuration tool olarak kullanılır. Yani önce infrastructure u oluşturursun. Sonra onu configure etmek için Ansible I kullanırsın.
- - Terraforma göre daha eskidir.

Terraform & Ansible

BENZERLİKLER

- İkisi de IAC tool'u olarak kullanılır. Yani ikisiyle de altyapıyı sağlarız, yapılandırırırz ve yönetiriz.

TERRAFORM

- 1) Core
- 2) State