Introducción a Containers

Container Engines - ECS

Mauricio Améndola / Sebastián Orrego – Profesor Adjunto
Escuela de Tecnología – Facultad de Ingeniería

UNIVERSIDAD ORT
Uruguay

AGENDA

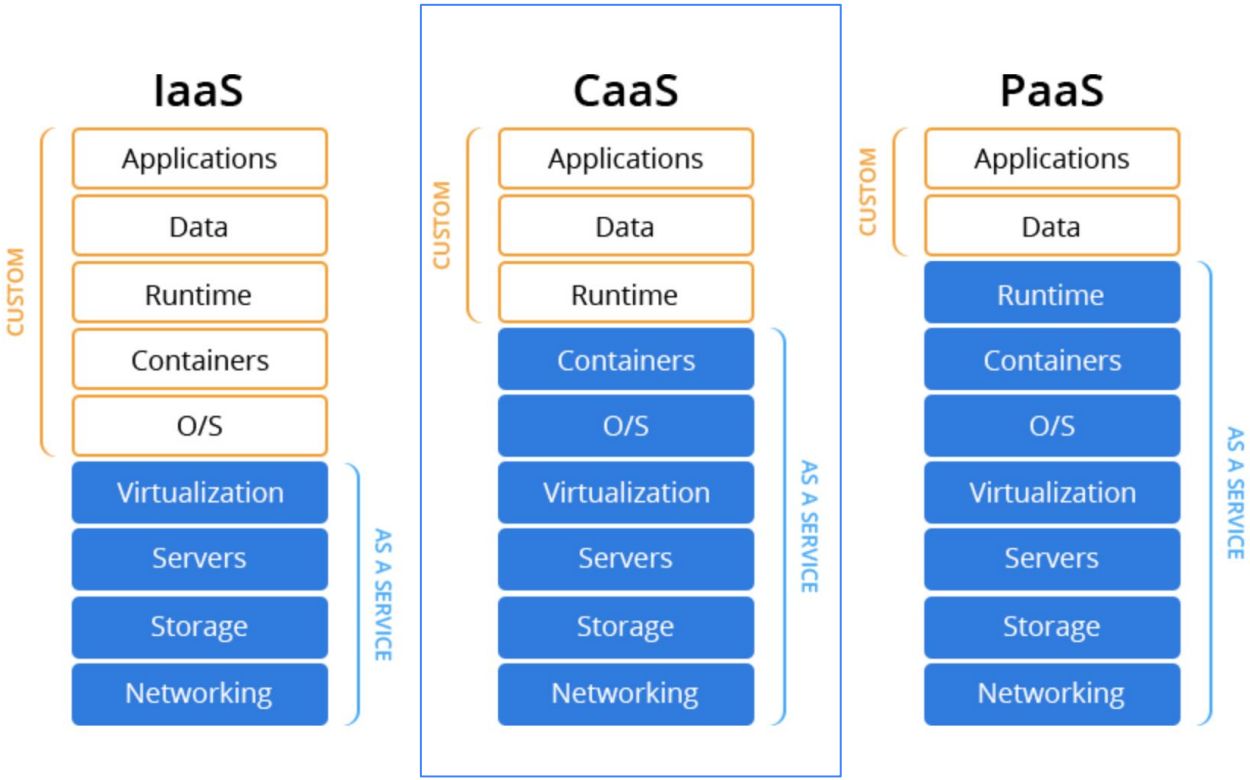
1. Elastic container Service
2. Fargate
3. ECS EC2 vs Fargate
4. Elastic container Registry
5. Copilot

Elastic Containers Services

ECS

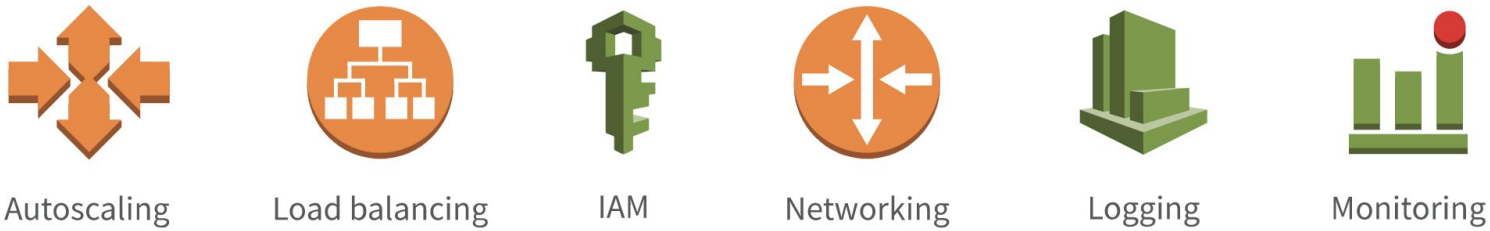
Servicio escalable de orquestación de contenedores completamente administrado por AWS.
Se utiliza para ejecutar aplicaciones de Docker en un clúster escalable.

AWS ECS elimina la necesidad de instalar y mantener servicios de containers propios.


IaaS
CUSTOM
Applications
Data
Runtime
Containers
O/S
Virtualization
Servers
Storage
Networking
AS A SERVICE
CaaS
CUSTOM
Applications
Data
Runtime
Containers
O/S
Virtualization
Servers
Storage
Networking
AS A SERVICE
PaaS
CUSTOM
Applications
Data
Runtime
Containers
O/S
Virtualization
Servers
Storage
Networking
AS A SERVICE

ECS

ECS además se integra con varios servicios de AWS para extender las capacidades y casos de uso.



ECS

Componentes:

- ➔ Cluster
- ➔ Container definition
- ➔ Task definition
- ➔ Service

ECS

Componentes:

➔ Cluster

Conjunto de instancias de contenedor que ejecutan el agente de contenedores de Amazon ECS.

Por definición un Cluster es un grupo lógico de servicios o tasks.

ECS

Componentes:

➔ Task definition & container definition

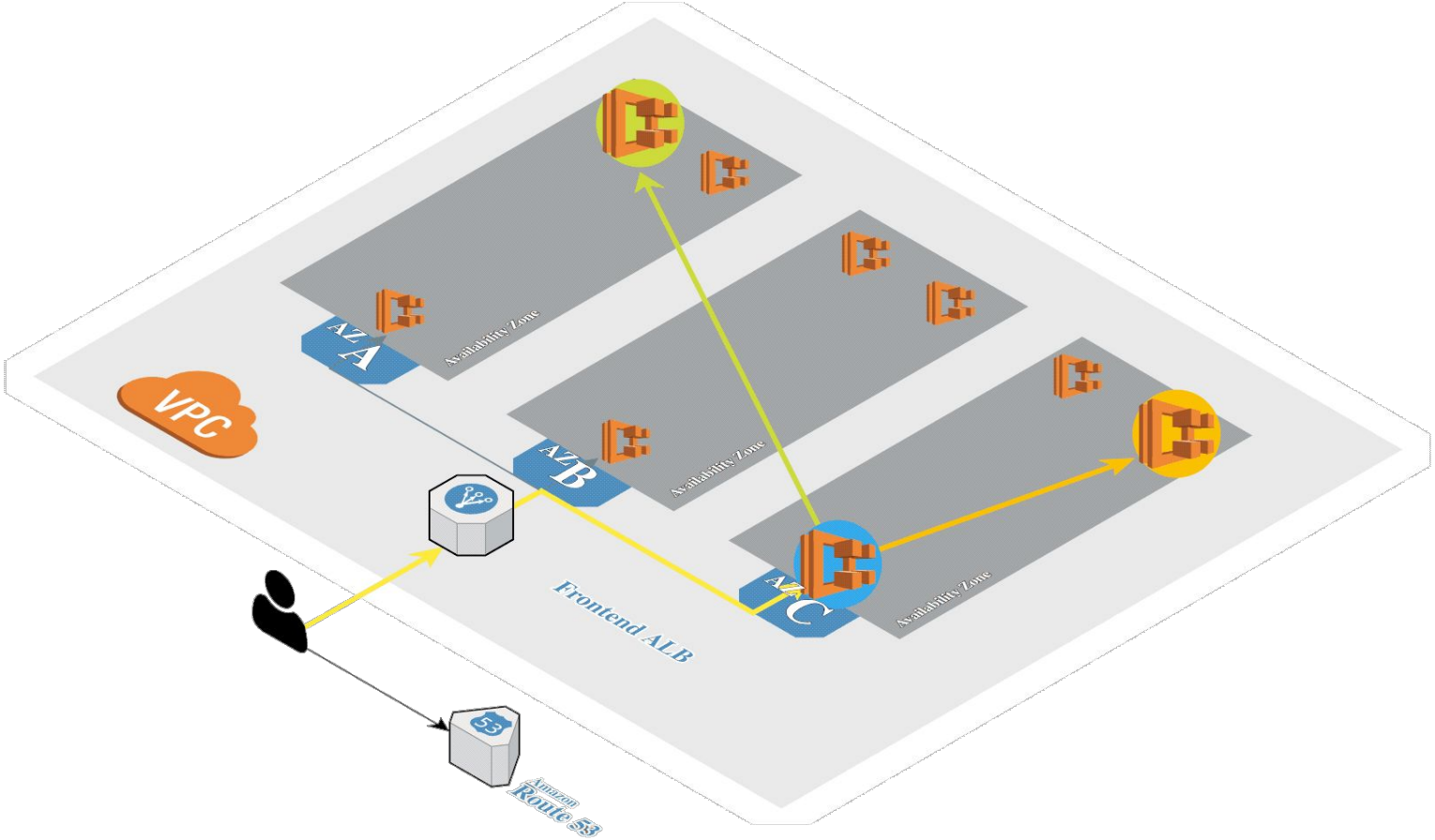
La definición de tarea tal cómo su nombre lo indica define el o los containers que se van a ejecutar dentro. Acá se especifican cosas cómo, la imagen, recursos de cómputo y networking. Dentro de una task definition, debemos de crear un ***container definition***.

ECS

Componentes:

➔ Services

AWS ECS permite correr un determinado número de instancias de una Task definition al mismo tiempo. Esto se conoce cómo ***Service***. Si una task se detiene por algún motivo, ECS se encarga de levantar una nueva instancia de la Task. Este trabajo le corresponde a un servicio llamado **Scheduler**.


VPC
AZ A
Availability Zone
AZ B
Availability Zone
AZ C
Availability Zone
Frontend ALB
53
Amazon
Route 53

Serverless way

Fargate

Existe otro modelo de despliegue compatible con ECS llamado Fargate. Esta tecnología de containers permite correr y escalar containers sin la necesidad de desplegar un cluster de instancias de EC2. En esta modalidad, los containers corren en servidores completamente gestionados por AWS sin interacción nuestra. Este modelo opera bajo el concepto de ***Serverless***.

ECS sobre EC2 o Fargate?

ECS EC2 vs Fargate

Técnicamente las dos opciones son más que válidas y ofrecen las mismas capacidades y calidad de servicio. Para tomar la decisión hay que considerar factores cómo el modelo de costos y el tipo de carga de trabajo.

ECS EC2 vs Fargate

El modelo de costos difiere en ambos servicios. Mientras que con EC2 se cobra el tiempo de ejecución de la instancia, en Fargate se cobra solo por el cómputo del container. En EC2 da lo mismo si tenemos 1 o 10 containers, mientras que en Fargate, el costo será por el total de containers.

ECS EC2 vs Fargate

Las cargas de trabajo pueden inclinar la balanza por uno u otro, por ejemplo, no es lo mismo una aplicación con consumo constante que una aplicación donde el consumo varía según la demanda o franjas horarias.

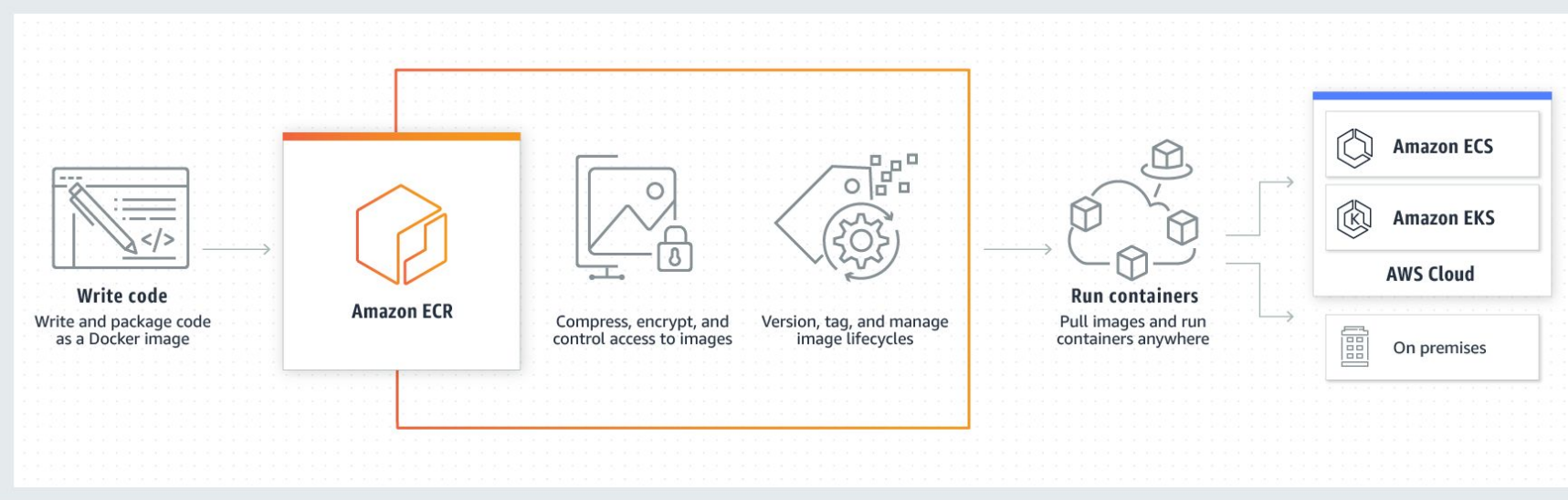

SHOW TIME!!!
imgflip.com

Elastic Container Registry

ECR

Servicio de Registro de contenedores completamente administrado que facilita a los desarrolladores las tareas de almacenamiento, administración e implementación de imágenes de contenedores.

ECR


Write code
Write and package code as a Docker image
Amazon ECR
Compress, encrypt, and control access to images
Version, tag, and manage image lifecycles
Run containers
Pull images and run containers anywhere
Amazon ECS
Amazon EKS
AWS Cloud
On premises

ECR

Es necesaria la cli de AWS para poder loguearse en la registry (existen otros mecanismos también).

$ aws ecr get-login-password —region us-east-1| docker login —username AWS —password-stdin ***url_registry_amazon***

The IaC way

IaC

Para trabajar con ECS y desplegar nuestros containers podemos usar Terraform o algún otro software similar.
Existe una herramienta, llamada ***copilot*** desarrollada por AWS para facilitar la creación y despliegue de ECS + containers.

Copilot

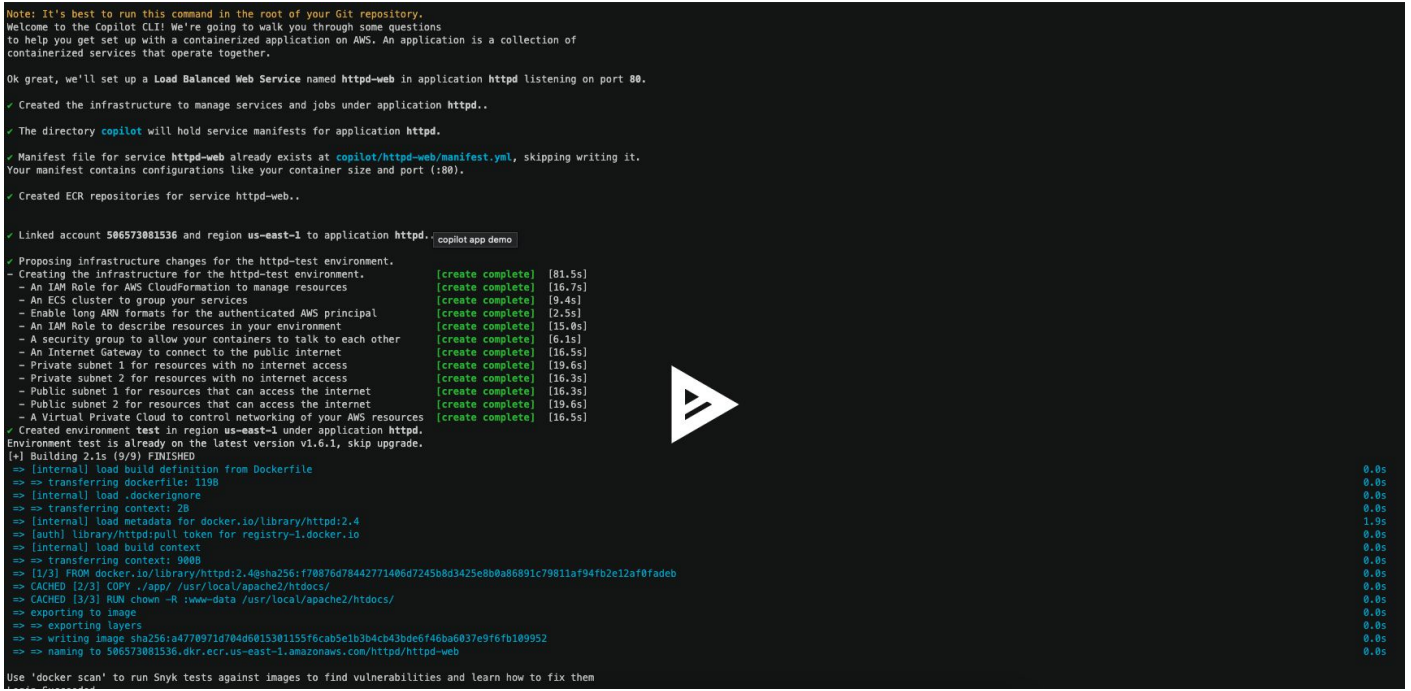
Copilot

Es un cliente de línea de comandos, compatible con MacOS, Linux y Windows capaz de buildear y desplegar containers sobre AWS Fargate y AWS App Runner.

Copilot se encarga de desplegar todos los recursos necesarios:

- ➔ Cluster de ECS, task definitions, services, etc
- ➔ Repositorios de ECR
- ➔ Subnets, VPCs, etc
- ➔ ALBs
- ➔ Etc.

Copilot-cli


Note: It's best to run this command in the root of your Git repository.
Welcome to the Copilot CLI! We're going to walk you through some questions
to help you get set up with a containerized application on AWS. An application is a collection of
containerized services that operate together.
Ok great, we'll set up a Load Balanced Web Service named httpd-web in application httpd listening on port 80.
✔ Created the infrastructure to manage services and jobs under application httpd..
✔ The directory copilot will hold service manifests for application httpd.
✔ Manifest file for service httpd-web already exists at copilot/httpd-web/manifest.yml, skipping writing it.
Your manifest contains configurations like your container size and port (:80).
✔ Created ECR repositories for service httpd-web..
✔ Linked account 506573081536 and region us-east-1 to application httpd.
copilot app demo
✔ Proposing infrastructure changes for the httpd-test environment.
– Creating the infrastructure for the httpd-test environment. [create complete] [81.5s]
– An IAM Role for AWS CloudFormation to manage resources [create complete] [16.7s]
– An ECS cluster to group your services [create complete] [9.4s]
– Enable long ARN formats for the authenticated AWS principal [create complete] [2.5s]
– An IAM Role to describe resources in your environment [create complete] [15.0s]
– A security group to allow your containers to talk to each other [create complete] [6.1s]
– An Internet Gateway to connect to the public internet [create complete] [16.5s]
– Private subnet 1 for resources with no internet access [create complete] [19.6s]
– Private subnet 2 for resources with no internet access [create complete] [16.3s]
– Public subnet 1 for resources that can access the internet [create complete] [16.3s]
– Public subnet 2 for resources that can access the internet [create complete] [19.6s]
– A Virtual Private Cloud to control networking of your AWS resources [create complete] [16.5s]
✔ Created environment test in region us-east-1 under application httpd.
Environment test is already on the latest version v1.6.1, skip upgrade.
[+] Building 2.1s (9/9) FINISHED
=> [internal] load build definition from Dockerfile 0.0s
=> => transferring dockerfile: 119B 0.0s
=> [internal] load .dockerignore 0.0s
=> => transferring context: 2B 0.0s
=> [internal] load metadata for docker.io/library/httpd:2.4 1.9s
=> [auth] library/httpd:pull token for registry-1.docker.io 0.0s
=> [internal] load build context 0.0s
=> => transferring context: 900B 0.0s
=> [1/3] FROM docker.io/library/httpd:2.4@sha256:f70876d78442771406d7245b8d3425e8b0a86891c79811af94fb2e12af0fadeb 0.0s
=> CACHED [2/3] COPY ./app/ /usr/local/apache2/htdocs/ 0.0s
=> CACHED [3/3] RUN chown -R :www-data /usr/local/apache2/htdocs/ 0.0s
=> exporting to image 0.0s
=> => exporting layers 0.0s
=> => writing image sha256:a4770971d704d6015301155f6cab5e1b3b4cb43bde6f46ba6037e9f6fb109952 0.0s
=> => naming to 506573081536.dkr.ecr.us-east-1.amazonaws.com/httpd/httpd-web 0.0s
Use 'docker scan' to run Snyk tests against images to find vulnerabilities and learn how to fix them


SHOW TIME!!!
imgflip.com