Wifi / Password

Wifi name:
TDPK-WIFI

Username:
AWSOn-boardingworkshop1
AWSOn-boardingworkshop2
AWSOn-boardingworkshop3
AWSOn-boardingworkshop4
AWSOn-boardingworkshop5

Password
Welcome@2022

https://github.com/TIDC-PS-Inter/AWS-Workshop

Part 1

AWS Workshop Series

Day 3: Container for Beginner

Taking Enterprise Beyond the Cloud by TrueIDC

Mr. Niran Sohinkong

Professional Service Manager

Presented by

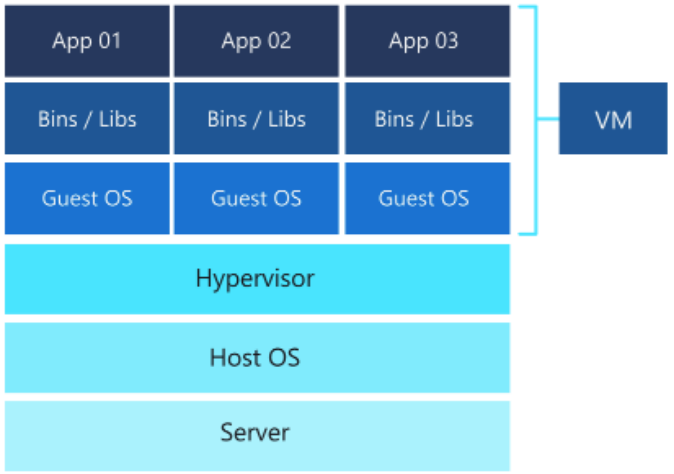
- Niran Sohinkong (Nueng)
- Professional Service Manager, TrueIDC
- AWS DevOps
- AWS SysOps / Architect
- niran.soh@ascendcorp.com

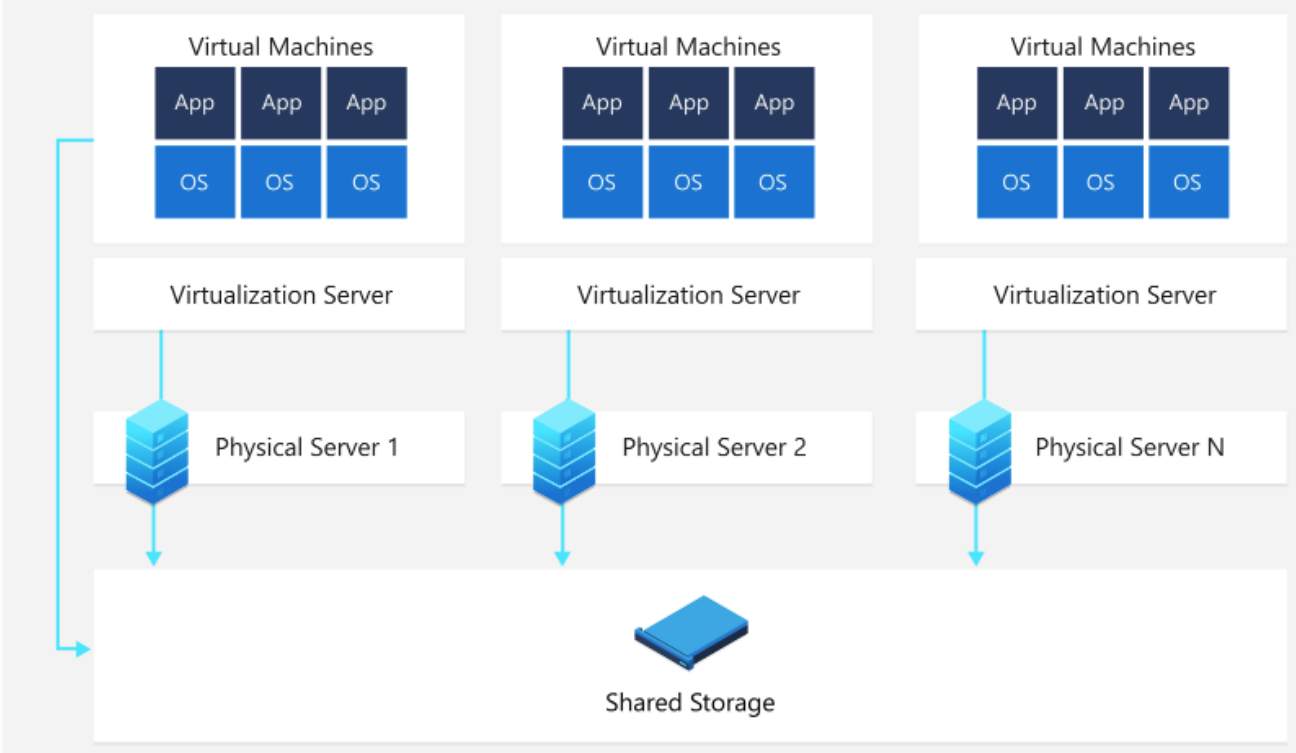
trueIDC

Agenda

- What is a Container?
- Containers and Virtual Machines
- What is Docker?
- Docker Architecture
- AWS ECS/ECR Overview
- Lab

Virtual Machines

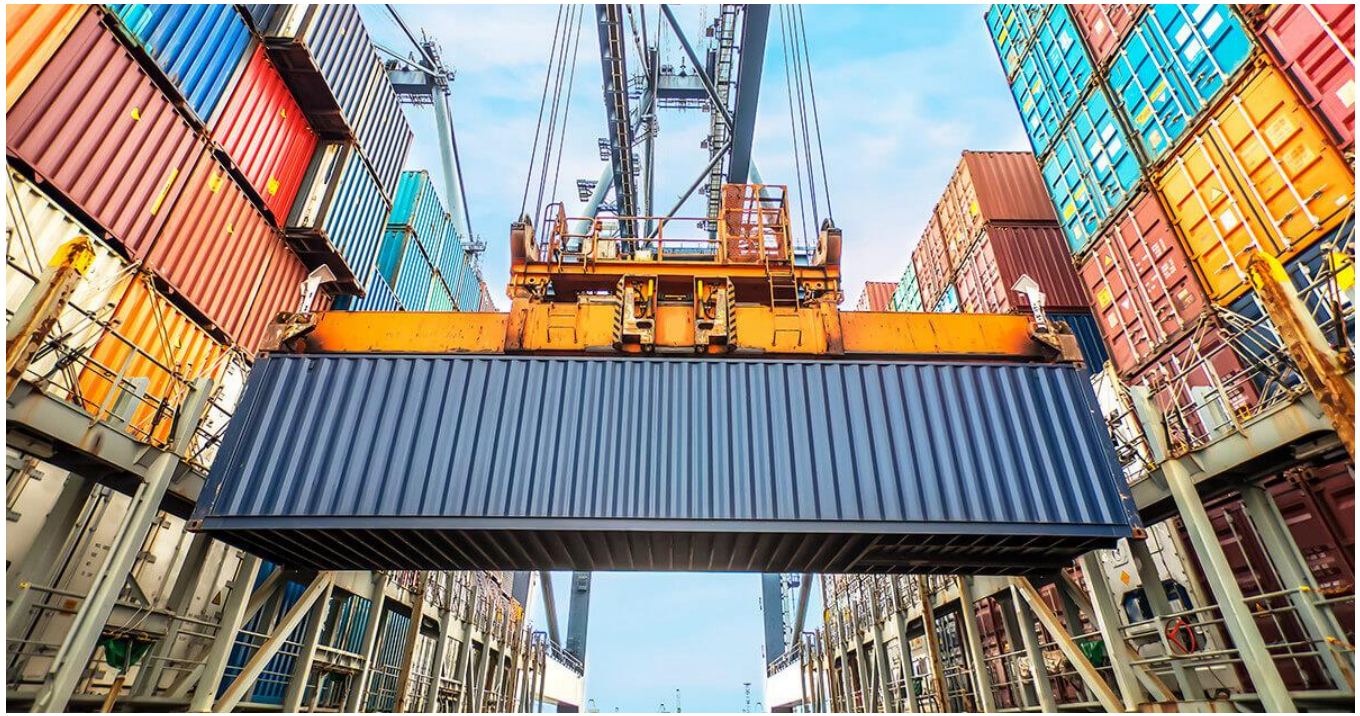




A virtual machine, commonly shortened to just VM, is no different than any other physical computer like a laptop or server. It has a CPU, memory, disks to store your files, and can connect to the internet if needed. While the parts that make up your computer (called hardware) are physical and tangible, VMs are often thought of as virtual computers or software-defined computers within physical servers, existing only as code.

Credit by: https://azure.microsoft.com/en-us/resources/cloud-computing-dictionary/what-is-a-virtual-machine/#overview

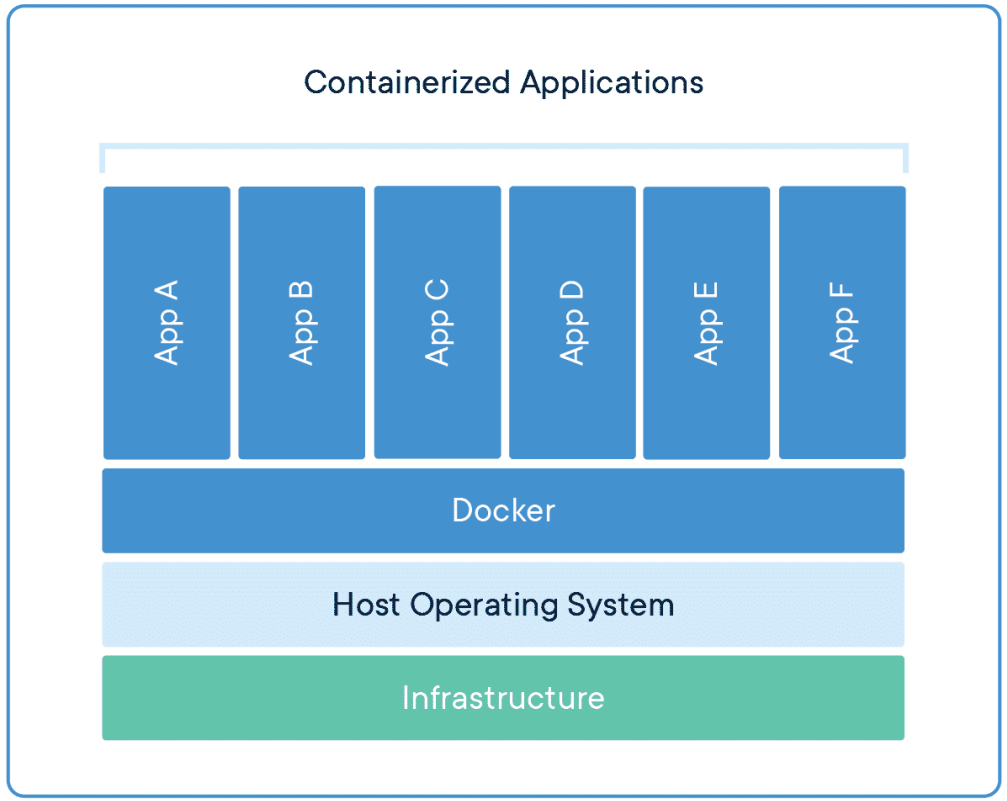
What is a Container?



A container is a standard unit of software that packages up code and all its dependencies, so the application runs quickly and reliably from one computing environment to another.

Credit by: https://www.docker.com/resources/what-container/
https://www.ionos.de/digitalguide/server/knowhow/docker-tools-das-oekosystem-der-container-plattform/

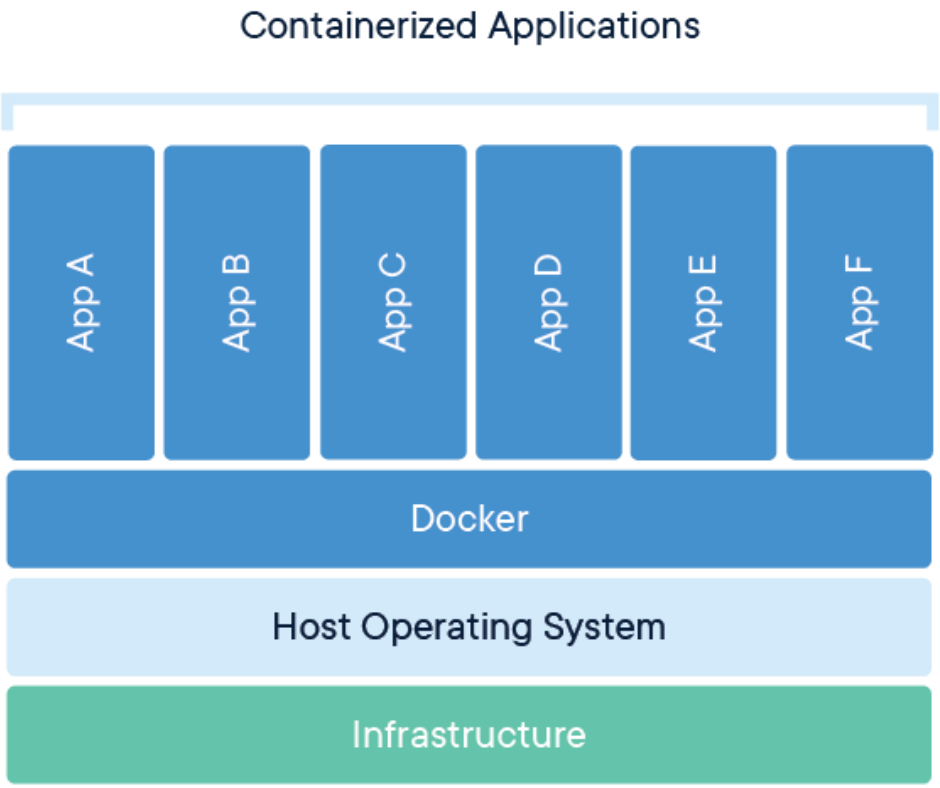
What is a Container?

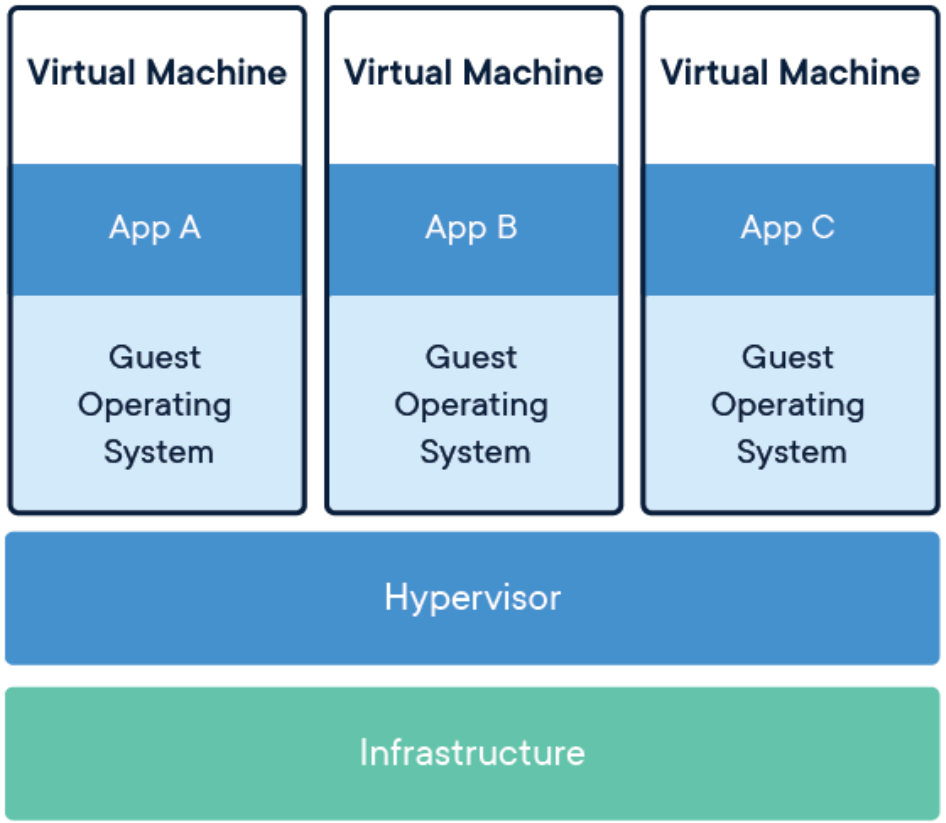


A container is a standard unit of software that packages up code and all its dependencies, so the application runs quickly and reliably from one computing environment to another.

Credit by: https://www.docker.com/resources/what-container/

Containers and Virtual Machines





Credit by: https://www.docker.com/resources/what-container/

Containers vs Virtual Machines

Comparison	Containers	Virtual Machines
Boot Time	ใช้เวลาไม่กี่วินาทีในการ Boot	ใช้เวลาเป็นนาทีในการ Boot
Execution	จัดการผ่าน Container Engine	จัดการผ่าน Hypervisor
Memory	ไม่เปลือง Memory เนื่องจากไม่ต้องใช้ในการสร้างระบบ Virtual	ใช้ Memory ทุกครั้งเนื่องจาก OS จะต้อง Start ก่อนที่จะใช้งาน Service
Isolation	ระบบไม่ได้แยกการทำงานออกจากกันชัดเจน อาจเกิดความยุ่งยากได้ง่าย	ระบบมีการแยกการทำงานออกจากกัน จัดการปัญหาได้ง่ายกว่า
Ease of Deployment	Deploy Container ทำได้ง่ายเนื่องจาก Container Image สามารถติดตั้งได้ในหลาย ๆ OS	Deploy VM มีขั้นตอนที่ต้องเตรียมการมากกว่าเพราะต้องแยก Instance ออกจากกัน

Ref: https://blog.cloudhm.co.th/containers-vs-vm/

trueIDC

What is Docker?

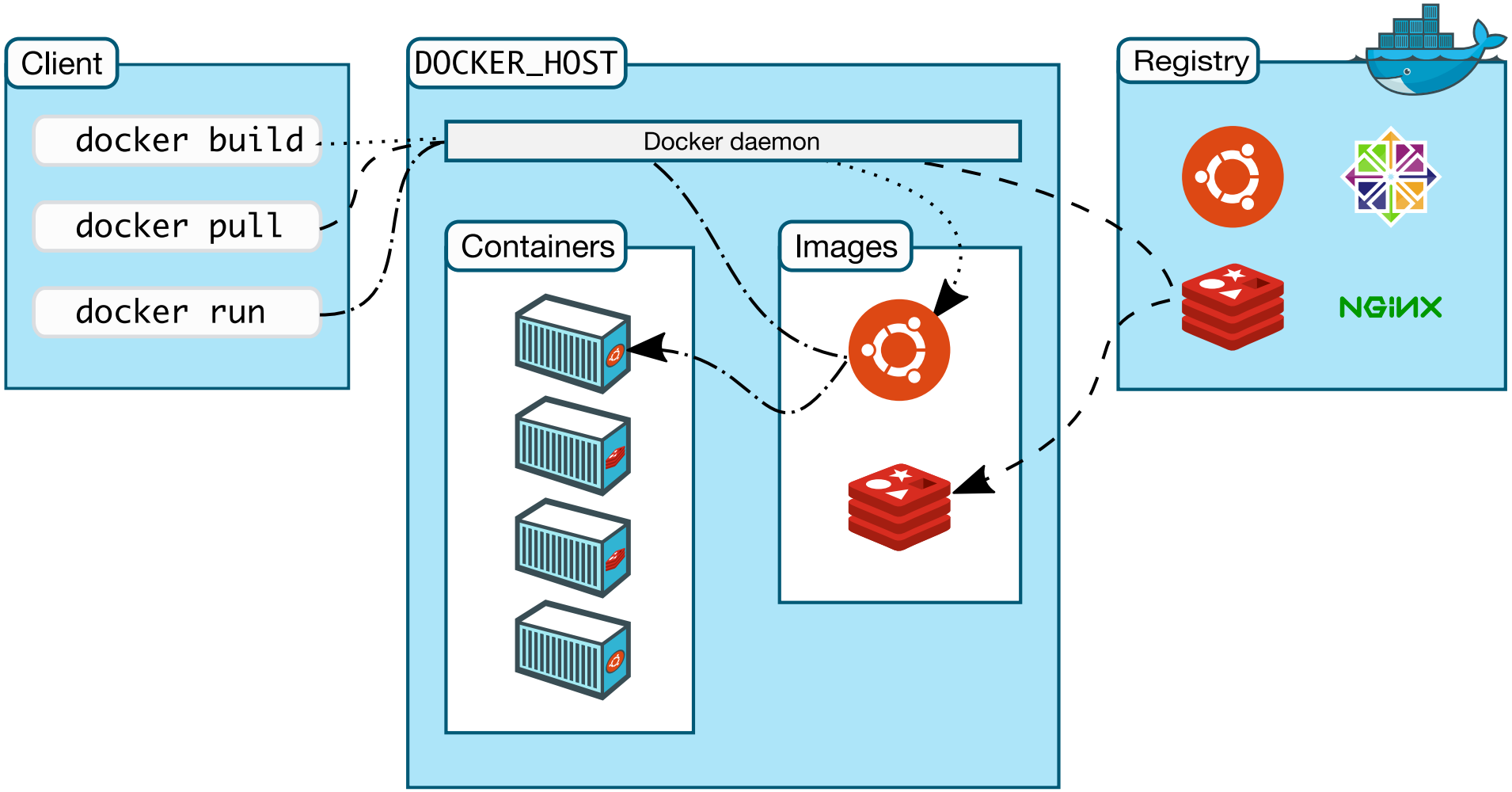
Docker provides the ability to package and run an application in a loosely isolated environment called a container. The isolation and security allows you to run many containers simultaneously on a given host.

Containers are lightweight and contain everything needed to run the application, so you do not need to rely on what is currently installed on the host.

Docker provides tooling and a platform to manage the lifecycle of your containers

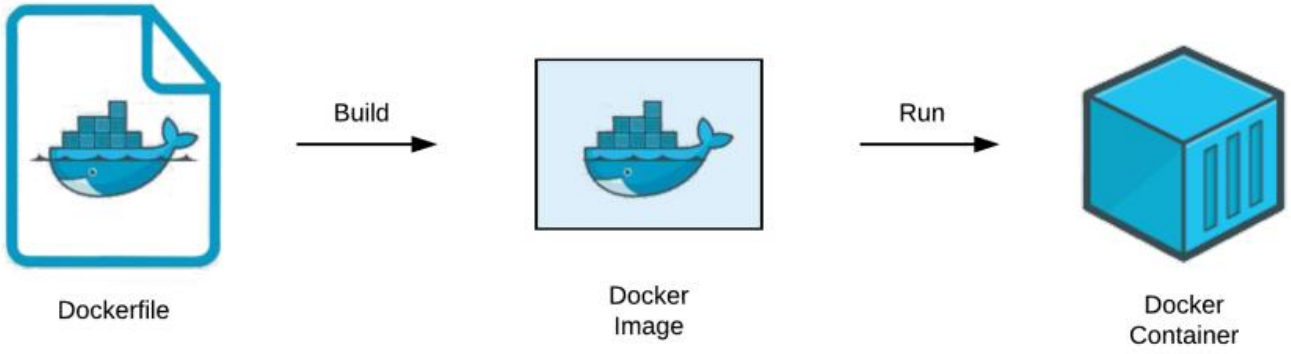
Credit by: https://docs.docker.com/get-started/overview/

Docker architecture



Credit by: https://docs.docker.com/get-started/overview/

Where the Docker Image come from?



- **Dockerfile** – is a text document that contains all the commands you would normally execute manually in order to build a Docker image. Docker can build images automatically by reading the instructions from a Dockerfile.
- **Docker Images** – are the basis of containers. An Image is an ordered collection of root filesystem changes and the corresponding execution parameters for use within a container runtime. An image typically contains a union of layered filesystems stacked on top of each other. An image does not have state and it never changes.
- **Docker Container**– is a runtime instance of a docker image.

Credit by: https://neilkillen.com/2020/02/28/introducing-docker-to-your-team/

Dockerfile

FROM node:16-alpine	---> using node with alipine image
WORKDIR /usr/src/app	---> Create app directory
COPY package*.json ./	---> Copy package to app directoty
RUN npm install	
COPY . .	---> Bundle app source
EXPOSE 3000	---> Listen port 3000
CMD ["node", "index.js"]	---> execute index.js

trueIDC

What are the benefits of Docker?

โดยสรุป **Docker Container** เกิดมาเพื่อช่วยในการทำงานเฉพาะส่วนของ **Application Level** ไม่ได้เอามาทั้ง **OS windows, driver** ต่าง ๆ ---> ซึ่งจะเลือกมาเฉพาะที่ใช้งานจริง ๆ

1. ทำให้ start, stop, auto scaling ได้เร็วกกว่า VM
2. Utilize CPU และ Mem น้อยกว่า VM
3. ไม่สิ้นเปลืองพื้นที่ในการจัดเก็บ
4. สามารถแยกออกเป็น แต่ละ env ได้ง่าย เช่น Dev, QA, UAT, SIT, NonProd, Prod ได้ง่าย สะดวก และรวดเร็ว
5. สามารถแยกแอพพลิเคชันยุคเก่าที่เปนก้อนใหญ่ๆแบบ Monolith มาเป็น Microservice ที่มีขนาดเล็กลง สามารถทำงานแยกกันได้อย่างอิสระ และ scaling service ได้ง่ายขึ้น

Credit by: https://medium.com/accesto/what-is-docker-and-why-to-use-it-explained-for-executives-b75fa1b2f516

Linux Command

#pwd (present working directory)
#cd (change directory)
#mkdir (create directory)
#rmdir (remove directory)
#touch (create file)
#ls
#which
#uname -a
#cat
#cat /etc/os-release
#netstat -an | grep ":80"

VI editor command
:q (quit)
:wq (save & quit)

More command from: https://www.hostinger.com/tutorials/linux-commands

Docker Command

#docker build
#docker images
#docker run
#docker ps
#docker pull
#docker push
#docker stop
#docker rmi

More details from: https://docs.docker.com/engine/reference/commandline/docker/

Docker Command - Example

Install Docker

```
[root@PSWWDDV2 docker]# yum install docker
Last metadata expiration check: 2:45:03 ago on Wed 20 Jul 2022 08:01:02 AM +07.
Dependencies resolved.
================================================================================================================================
 Package                              Architecture   Version                                          Repository         Size
================================================================================================================================
Installing:
 podman-docker                        noarch         3.3.1-9.module_el8.5.0+988+b1f0b741              appstream          56 k
Installing dependencies:
 conmon                               x86_64         2:2.0.29-1.module_el8.5.0+890+6b136101           appstream          52 k
 container-selinux                    noarch         2:2.167.0-1.module_el8.5.0+911+f19012f9          appstream          54 k
 containernetworking-plugins          x86_64         1.0.0-1.module_el8.5.0+890+6b136101              appstream          19 M
 containers-common                    noarch         2:1-2.module_el8.5.0+890+6b136101                appstream          79 k
 criu                                 x86_64         3.15-3.module_el8.5.0+890+6b136101               appstream         518 k
 fuse-common                          x86_64         3.2.1-12.el8                                     baseos             21 k
 fuse-overlayfs                       x86_64         1.7.1-1.module_el8.5.0+890+6b136101              appstream          73 k
 fuse3                                x86_64         3.2.1-12.el8                                     baseos             50 k
 fuse3-libs                           x86_64         3.2.1-12.el8                                     baseos             94 k
 iptables                             x86_64         1.8.4-20.el8                                     baseos            585 k
 libnet                               x86_64         1.1.6-15.el8                                     appstream          67 k
 libnetfilter_conntrack               x86_64         1.0.6-5.el8                                      baseos             65 k
 libnfnetlink                         x86_64         1.0.1-13.el8                                     baseos             33 k
 libnftnl                             x86_64         1.1.5-4.el8                                      baseos             83 k
 libslirp                             x86_64         4.4.0-1.module_el8.5.0+890+6b136101              appstream          70 k
 nftables                             x86_64         1:0.9.3-21.el8                                   baseos            321 k
 podman                               x86_64         3.3.1-9.module_el8.5.0+988+b1f0b741              appstream          12 M
 podman-catatonit                     x86_64         3.3.1-9.module_el8.5.0+988+b1f0b741              appstream         340 k
 protobuf-c                           x86_64         1.3.0-6.el8                                      appstream          37 k
 runc                                 x86_64         1.0.2-1.module_el8.5.0+911+f19012f9              appstream         3.1 M
 slirp4netns                          x86_64         1.1.8-1.module_el8.5.0+890+6b136101              appstream          51 k
Enabling module streams:
 container-tools                                     rhel8

Transaction Summary
================================================================================================================================
Install  22 Packages

Total download size: 37 M
Installed size: 127 M
Is this ok [y/N]:
```

Docker Command - Example

Docker build image

```
[root@PSWWDDV2 nginx]# docker build -t web:v1 .
Emulate Docker CLI using podman. Create /etc/containers/nodocker to quiet msg.
STEP 1/4: FROM httpd:2.4
✓ docker.io/library/httpd:2.4
Trying to pull docker.io/library/httpd:2.4...
Getting image source signatures
Copying blob 461246efe0a7 skipped: already exists
Copying blob d6bc17b4451a done
Copying blob 97f4b88189d8 done
Copying blob c332ae8365a7 done
Copying blob 72dcd3e40e39 done
Copying config 444f7df01c done
Writing manifest to image destination
Storing signatures
STEP 2/4: WORKDIR /usr/local/apache2/htdocs/
--> 4ed69169fd4
STEP 3/4: COPY . .
--> 38c0bbd3f89
STEP 4/4: EXPOSE 80
COMMIT web:v1
--> 6dacb91dad3
Successfully tagged localhost/web:v1
6dacb91dad355a3751147ce5e7de89ae7a438d78e2ff01e9c5652341158508fc
```

Docker images

```
[root@PSWWDDV2 nginx]# docker images
Emulate Docker CLI using podman. Create /etc/containers/nodocker to quiet msg.
REPOSITORY                TAG         IMAGE ID      CREATED             SIZE
localhost/web             v1          6dacb91dad35  10 seconds ago      149 MB
localhost/api-app         v1          9e76e1a46c9f  About an hour ago   122 MB
docker.io/library/nginx   latest      670dcc86b69d  14 hours ago        146 MB
docker.io/library/node    16-alpine   b0cbdedc1b9d  33 hours ago        115 MB
docker.io/library/httpd   2.4         444f7df01ce9  8 days ago          149 MB
```

trueIDC

Docker Command - Example

Docker run container

```
[root@PSWWDDV2 nginx]# docker run -d -p 80:80 web:v1
Emulate Docker CLI using podman. Create /etc/containers/nodocker to quiet msg.
5ba3283cfa8d66c3ae8cfedcad76201017405be33a0b1ee2a47ba5661c138c7c
```

Docker ps

```
[root@PSWWDDV2 nginx]# docker ps
Emulate Docker CLI using podman. Create /etc/containers/nodocker to quiet msg.
CONTAINER ID  IMAGE                 COMMAND           CREATED            STATUS                 PORTS                   NAMES
cb2b38e1a27b  localhost/api-app:v1  node index.js     About an hour ago  Up About an hour ago   0.0.0.0:3000->3000/tcp  sad_edison
5ba3283cfa8d  localhost/web:v1      httpd-foreground  30 seconds ago     Up 30 seconds ago      0.0.0.0:80->80/tcp      ecstatic_swirles
```

Introduction to Amazon ECS and AWS Fargate

Containers Immersion Day:

AWS container services landscape

Management

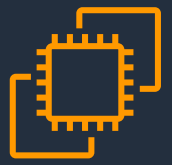
Deployment, Scheduling, Scaling & Management of containerized applications

Amazon Elastic Container Service

Amazon Elastic Kubernetes Service

Hosting

Where the containers run



Amazon EC2

AWS Fargate

ECS Anywhere

EKS Anywhere

Image Registry

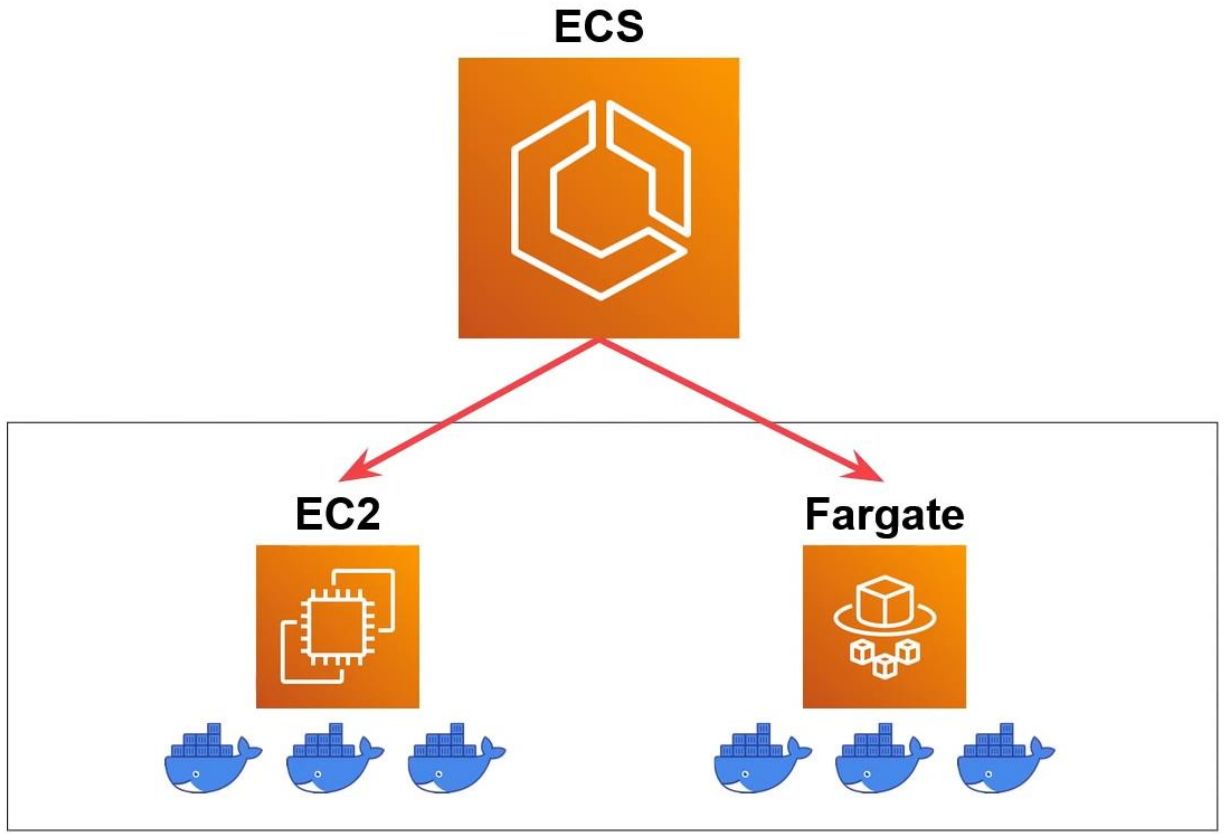
Container Image Repository

Amazon Elastic Container Registry

Amazon Elastic Container Service

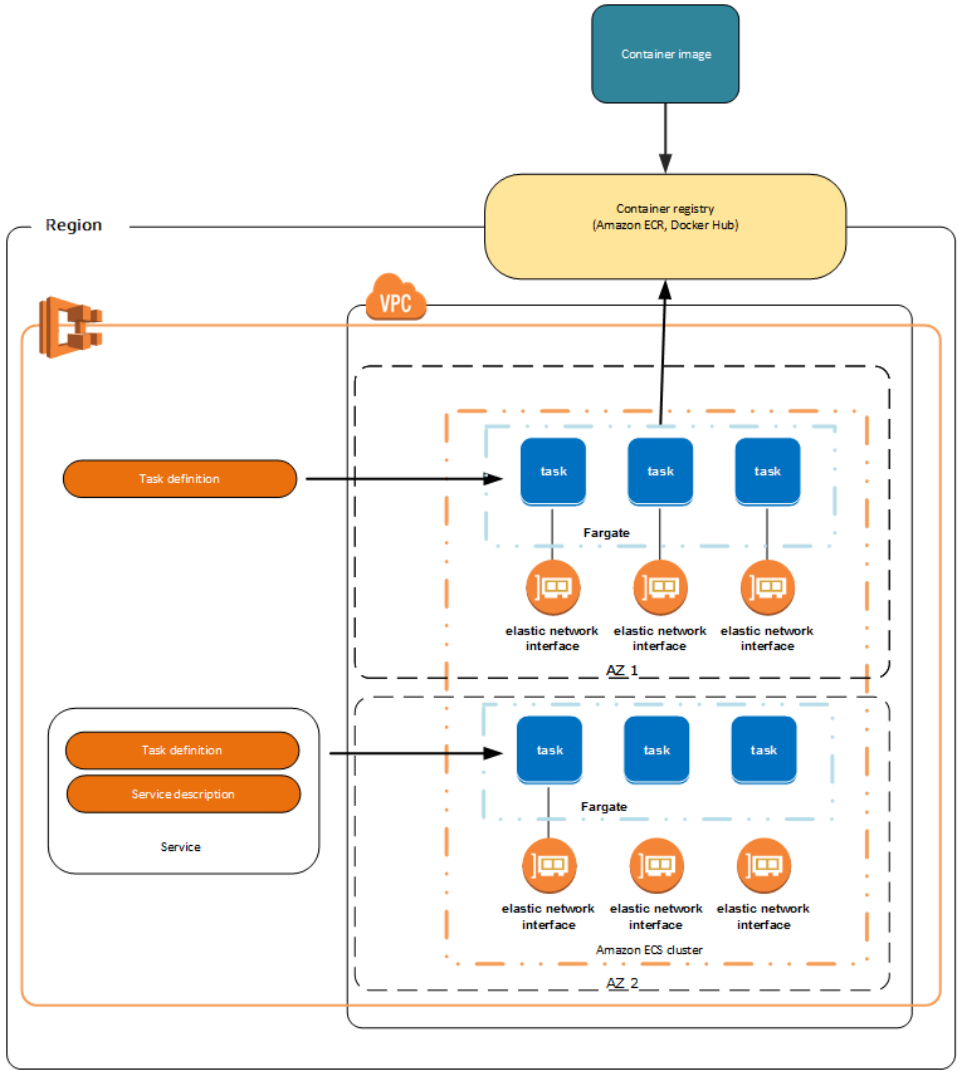
AWS ECS

Amazon Elastic Container Service (Amazon ECS) is a highly scalable and fast container management service. You can use it to run, stop, and manage containers on a cluster.



https://docs.aws.amazon.com/AmazonECS/latest/developerguide/Welcome.html

AWS ECS Overview



Tasks

A *task* is the instantiation of a task definition within a cluster.

Task definitions

A *task definition* is a text file that describes one or more containers that form your application. The task definition functions as a blueprint for your application.

Services

To run, maintain and manage your desired number of tasks based on your task definition.

Container image

Images are typically built from a Dockerfile. A Dockerfile is a plaintext file that specifies all of the components that are included in the container.

trueIDC

Introduction to Amazon Elastic Container Registry

Containers Immersion Day: Module 2

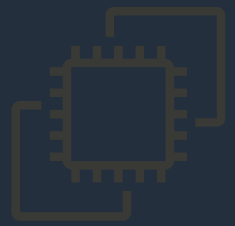
AWS container services landscape

Management
Deployment, Scheduling, Scaling & Management of containerized applications

Amazon Elastic Container Service

Amazon Elastic Container Service for Kubernetes

Hosting
Where the containers run



Amazon EC2

AWS Fargate

Image Registry
Container Image Repository

Amazon Elastic Container Registry

The benefits of using Amazon ECR

Fully Managed

Secure

Highly Available

Simplified Workflow

AWS ECR

aws | Services | Search for services, features, blogs, docs, and more [Alt+S] | Singapore | AdministratorAccess/warin.tra@ascendcorp.com

Amazon Elastic Container Registry

- Private registry
- Public registry
- Repositories
  - Summary
  - Images
  - Permissions
  - Lifecycle Policy
  - Tags

- Getting started
- Documentation
- Public gallery

Amazon ECR > Repositories > awshisnonprodrepostg

awshisnonprodrepostg

View push commands | Edit

Images (74)

Delete | Scan

Find images

1

Image tag	Artifact type	Pushed at	Size (MB)	Image URI	Digest	Scan status	Vulnerabilities
api-staging-V13.6	Image	June 09, 2022, 12:09:26 (UTC+07)	119.49	Copy URI	sha256:062a116cafe1eb53579259baf91913...	-	-
web-staging-02062022	Image	June 02, 2022, 17:47:42 (UTC+07)	76.77	Copy URI	sha256:cbabf60f20873b4d44a6247884ef7a...	-	-
api-staging-02062022	Image	June 02, 2022, 17:46:11 (UTC+07)	119.49	Copy URI	sha256:a829aec8afdd8b8a1a338869e5abe6...	-	-
api-staging-29052022	Image	May 29, 2022, 11:34:00 (UTC+07)	119.16	Copy URI	sha256:4bc375fba3553884eda73b96de46a...	-	-
web-staging-29052022	Image	May 29, 2022, 11:32:14 (UTC+07)	76.71	Copy URI	sha256:2678b9122c9ed7fea936969d38f278...	-	-
api-staging-28052022	Image	May 28, 2022, 19:08:42 (UTC+07)	119.45	Copy URI	sha256:490b452fde95ea8890616f8e18aada...	-	-
api-staging-27052022	Image	May 27, 2022, 19:55:34 (UTC+07)	119.45	Copy URI	sha256:05ebe8acc65f34e417cf6bdf220ff58...	-	-

AWS Fargate

Your containerized applications

Managed by AWS

No EC2 Instances to provision, scale or manage

Elastic

Scale up & down seamlessly. Pay only for what you use

Integrated

With the AWS ecosystem: VPC Networking, Elastic Load Balancing, IAM Permissions, CloudWatch and more

AWS Fargate

Amazon Elastic Kubernetes Service

AWS Fargate

Amazon Elastic Container Service

- Fargate still belongs to either ECS or EKS
  - ECS/EKS concepts still apply
  - No EC2 (AMI) to manage
- Max 4 vCPU, 30 GB RAM
- No GPU
- No windows container
  - Convert to .Net core to run in Linux
- No daemon set, no privileged mode
- Cost based on CPU, memory, duration
- Same application container

AWS EKS vs ECS

Amazon Elastic Kubernetes Service

Amazon Elastic Container Service

- Some interchangeable concepts
  - Pod ~ Task
  - Replicaset + Service ~ Service
  - HPA ~ Task auto scaler
  - Cluster autoscaler ~ Capacity provider scaling
- Security
  - Static, dynamic scanning, AMI hardening, OPA concepts
- Both works with EFS
- Same application container

Task Definition

aws | Services | Search for services, features, blogs, docs, and more [Alt+S] | Singapore | AdministratorAccess/warin.tra@ascendcorp.com

New ECS Experience
Tell us what you think

Amazon ECS
- Clusters
- Task Definitions
- Account Settings

Amazon EKS
- Clusters

Amazon ECR
- Repositories

AWS Marketplace
- Discover software
- Subscriptions

Task Definitions

Task definitions specify the container information for your application, such as how many containers are part of your task, what resources they will use, how they are linked together, and which host ports they will use. Learn more

Create new Task Definition | Create new revision | Actions

Last updated on June 9, 2022 5:19:12 PM (0m ago)

Status: ACTIVE INACTIVE

Filter in this page

1-37 Page size 50

Task Definition	Latest revision status
api-gemo-staging	ACTIVE
api-master	ACTIVE
api-master-staging	ACTIVE
api-staging	ACTIVE
api-test	ACTIVE
apimaster	ACTIVE
apistaging	ACTIVE
apitest	ACTIVE
jasper-gemo-staging	ACTIVE
jasper-master	ACTIVE
jasper-master-staging	ACTIVE
jasper-staging	ACTIVE
jasper-test	ACTIVE

Task Definition

aws | Services | Search for services, features, blogs, docs, and more [Alt+S] | Singapore | AdministratorAccess/warin.tra@ascendcorp.com

New ECS Experience
Tell us what you think

Amazon ECS
- Clusters
- Task Definitions
- Account Settings

Amazon EKS
- Clusters

Amazon ECR
- Repositories

AWS Marketplace
- Discover software
- Subscriptions

Task Definitions > api-staging > status > ACTIVE

Task definition name : api-staging

Select a revision for more details

Create new revision | Actions

Last updated on June 9, 2022 5:19:58 PM (1m ago)

Status: Active | Inactive | 1 selected

Filter in this page

1-33 | Page size 50

Task Definition Name : Revision	Status
api-staging:38	Active
api-staging:37	Active
api-staging:36	Active
api-staging:35	Active
api-staging:34	Active
api-staging:33	Active
api-staging:32	Active
api-staging:31	Active
api-staging:30	Active
api-staging:29	Active
api-staging:28	Active

trueIDC

Task Definition

aws Services Search for services, features, blogs, docs, and more

Task memory maximum allocation for container memory re

0

Container definitions

Add container

Container Name	Image
api-staging	1363634

Constraint

Task placement constraints allow you to filter the container insta
strategy to place the task.

Type

Add constraint

Service integration

AWS App Mesh is a service mesh based on the Envoy proxy tha
applications. To enable App Mesh integration, complete the follo

Edit container

Standard

Container name* api-staging

Image* 136363414653.dkr.ecr.ap-southeast-1.amazonaws.com/awshisnonprodrepostg:api-staging-02062022

Private repository authentication*

Memory Limits (MiB)* Hard limit 800

Soft limit 256

Define hard and/or soft memory limits in MiB for your container. Hard and soft limits correspond to the `memory` and `memoryReservation` parameters, respectively, in task definitions.
ECS recommends 300-500 MiB as a starting point for web applications.

Port mappings

Host port	Container port	Protocol
8061	80	tcp

trueIDC

Cluster -> Services

aws | Services | Search for services, features, blogs, docs, and more [Alt+S] | Singapore | AdministratorAccess/warin.tra@ascendcorp.com

New ECS Experience
Tell us what you think

Amazon ECS
- Clusters
- Task Definitions
- Account Settings

Amazon EKS
- Clusters

Amazon ECR
- Repositories

AWS Marketplace
- Discover software
- Subscriptions

Clusters > AWSHISNONPRDECS1003

Cluster : AWSHISNONPRDECS1003

Update Cluster | Delete Cluster

Get a detailed view of the resources on your cluster.

Cluster ARN	arn:aws:ecs:ap-southeast-1:136363414653:cluster/AWSHISNONPRDECS1003
Status	ACTIVE
Registered container instances	2
Pending tasks count	0 Fargate, 0 EC2, 0 External
Running tasks count	0 Fargate, 14 EC2, 0 External
Active service count	0 Fargate, 26 EC2, 0 External
Draining service count	0 Fargate, 0 EC2, 0 External

Services | Tasks | ECS Instances | Metrics | Scheduled Tasks | Tags | Capacity Providers

Create | Update | Delete | Actions

Last updated on June 9, 2022 5:24:52 PM (1m ago)

Filter in this page | Launch type ALL | Service type ALL | 1-26

Service Name	Status	Service type	Task Definition	Desired tasks	Running tasks	Launch type	Platform version
medcer-gemo-staging	ACTIVE	REPLICA	medcer-gemo-staging:1	1	1	EC2	--
patientimage	ACTIVE	REPLICA	patientimage01:7	1	1	EC2	--
web-gemo-staging	ACTIVE	REPLICA	web-gemo-staging:2	1	1	EC2	--
webmaster	ACTIVE	REPLICA	web-master:2	0	0	EC2	--
apitest	ACTIVE	REPLICA	api-test:1	0	0	EC2	--
signalr-master-staging	ACTIVE	REPLICA	signalr-master-staging:1	0	0	EC2	--
signalrtestUpdate	ACTIVE	REPLICA	signalr-test:1	0	0	EC2	--
jasper-gemo-staging	ACTIVE	REPLICA	jasper-gemo-staging:2	1	1	EC2	--

Cluster -> Services

aws | Services | Search for services, features, blogs, docs, and more [Alt+S] | Singapore | AdministratorAccess/warin.tra@ascendcorp.com

New ECS Experience
Tell us what you think

Amazon ECS
- Clusters
- Task Definitions
- Account Settings

Amazon EKS
- Clusters

Amazon ECR
- Repositories

AWS Marketplace
- Discover software
- Subscriptions

Clusters > AWSHISNONPRDECS1003 > Service: apistaging

Service : apistaging

Update | Delete

Cluster	AWSHISNONPRDECS1003	Desired count	2
Status	ACTIVE	Pending count	0
Task definition	api-staging:38	Running count	2
Service type	REPLICA		
Launch type	EC2		
Service role	ecsServiceRole		
Created By			

arn:aws:iam::136363414653:role/aws-reserved/sso.amazonaws.com/ap-southeast-1/AWSReservedSSO_AdministratorAccess_8d8a65e91274b7b6

Details | Tasks | Events | Auto Scaling | Deployments | Metrics | Tags | Logs

Load Balancing

Target Group Name	Container Name	Container Port
staging-api-8061	api-staging	80

Network Access

Health check grace period 0

trueIDC

Cluster -> Services

aws | Services | Search for services, features, blogs, docs, and more [Alt+S] | Singapore | AdministratorAccess/warin.tra@ascendcorp.com

Update Service

- **Step 1: Configure service**
- Step 2: Configure network
- Step 3: Set Auto Scaling (optional)
- Step 4: Review

Configure service

A service lets you specify how many copies of your task definition to run and maintain in a cluster. You can optionally use an Elastic Load Balancing load balancer to distribute incoming traffic to containers in your service. Amazon ECS maintains that number of tasks and coordinates task scheduling with the load balancer. You can also optionally use Service Auto Scaling to adjust the number of tasks in your service.

Task Definition
Family: api-staging
Enter a value
Revision: 38 (latest)

Launch type EC2

Switch to capacity provider strategy

Force new deployment ☐

Cluster AWSHISNONPRDECS1003

Service name apistaging

Service type* REPLICA

trueIDC

trueIDC

Lab: AWS Container Immersion Day

trueIDC | North Muangthong

trueIDC

Thank you

trueIDC | North Muangthong