

AWS Lab : Déployer WordPress sur Fargate



Dans ce deuxième lab, nous prenons l'image **WordPress déjà dans ECR** et la déployons sur **AWS ECS Fargate**.

- Créer un **cluster Fargate** sans serveur EC2.
- Définir une **Task Definition** et configurer le container WordPress.
- Lancer une **task Fargate** accessible depuis Internet.
- Accéder à WordPress dans votre navigateur pour vérifier le déploiement.

💡 Ce lab complète le premier (Docker → ECR) et montre comment passer de l'image stockée à une application **fonctionnelle sur Fargate**, étape par étape.

Sommaire

AWS Lab : Déployer WordPress sur Fargate	1
1 Requis — Exercice ECS Fargate COMPLET	2
2 — Crée un cluster ECS Fargate.....	2
3 — Crée une Task Definition (Fargate)	3
4 — Lancer une Task Fargate.....	3

5 — Accéder à WordPress.....	5
5.1 Cliquer sur la task en cours	5
5.2 Tester dans le navigateur Windows	5
6 — Vérifications	7
6.1 Vérifications CLI.....	7
6.2 vérifications via la console AWS Fargate	8

1 Requis — Exercice ECS Fargate COMPLET

Avoir une image (container) dans ECR

On part du principe que :

L'image WordPress est déjà dans ECR

On a l'URL ECR du lab précédent

Par exemple : (avec 123456789012 qui est l'ID de votre compte AWS)

123456789012.dkr.ecr.ca-central-1.amazonaws.com/wordpress:latest

2 — Créer un cluster ECS Fargate

AWS Console → ECS

Clusters

Create cluster

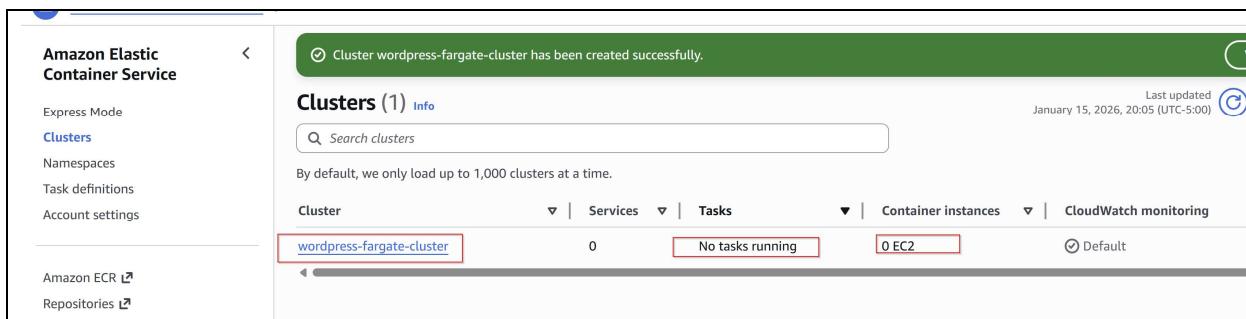
Paramètres :

Cluster name : wordpress-fargate-cluster

Infrastructure : AWS Fargate

Clique Create

Un cluster Fargate = logique seulement, pas de serveurs EC2.



3 — Crée une Task Definition (Fargate)

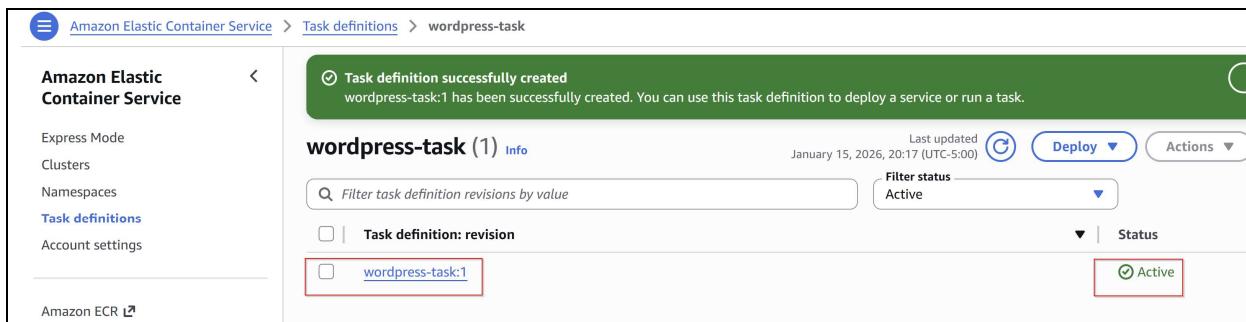
ECS → Task definitions
Create new task definition
Type : Fargate

Configuration :

Task definition name : wordpress-task
OS : Linux
CPU : 0.5 vCPU
Memory : 1 GB
Task role : none
Execution role : ecsTaskExecutionRole
(AWS le crée si besoin)

Ajouter le container
Container details :
Name : wordpress
Image URI :
123456789012.dkr.ecr.ca-central-1.amazonaws.com/wordpress:latest
Port mapping :
Container port : 80
Protocol : TCP
Clique Add container
Puis Create

On obtient l'image ci-dessous : (dans la console AWS ECS)



The screenshot shows the AWS ECS Task Definitions page. The left sidebar has 'Task definitions' selected. The main area displays a success message: 'Task definition successfully created' and 'wordpress-task:1 has been successfully created. You can use this task definition to deploy a service or run a task.' Below this, the 'wordpress-task (1)' section is shown with an 'Info' link. The task definition is listed as 'Active'. A red box highlights the 'Active' status in the status column.

4 — Lancer une Task Fargate

Aller dans Clusters
Cliquer wordpress-fargate-cluster
Onglet Tasks
Run new task

Paramètres :

Launch type : Fargate
Task definition : wordpress-task
Revision : latest
Number of tasks : 1

Réseau :

VPC : default
Subnet : public
Auto-assign public IP : ENABLED
Security group :
Autoriser HTTP (80) depuis 0.0.0.0/0

Pour le subnet, choisir un subnet public (vérifier que la table de routage du subnet contient une route qui permet le routage du traffic : 0.0.0.0 vers l'internet Gateway (IGW) du VPC.

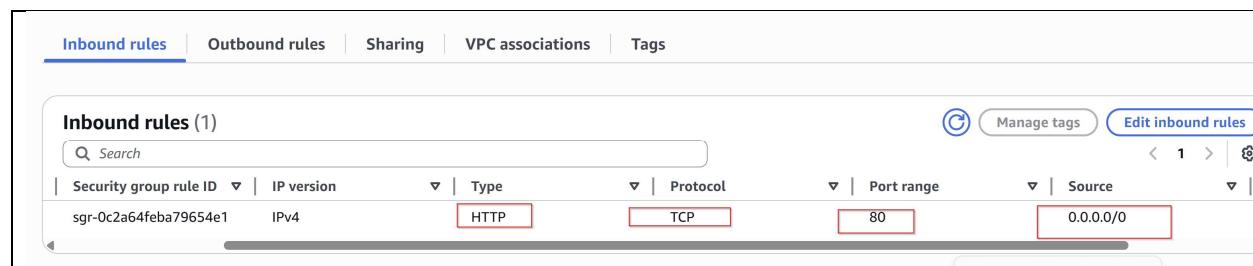
Pour le groupe de sécurité, choisir un groupe avec une règle en inbound, configurée adéquatement.
Pour voir les groupes de sécurité existants ou bien créer un nouveau groupe aller dans la console AWS EC2 – Security groups



The screenshot shows the AWS EC2 Security Groups interface. In the top navigation bar, 'EC2' is selected, followed by 'Security Groups'. Below this, a breadcrumb trail shows 'sg-0764192be43fcc533 - WebServerSG'. On the left, there's a sidebar with links like 'Savings Plans', 'Reserved Instances', 'Dedicated Hosts', 'Capacity Reservations', and 'Capacity Manager'. The main content area has a green header bar with the text 'Inbound security group rules successfully modified on security group (sg-0764192be43fcc533 | WebServerSG)' and a 'Details' button. Below this, the security group name 'sg-0764192be43fcc533 - WebServerSG' is displayed.

Règle que l'on doit avoir :

Ou bien restreindre la règle, en enlevant 0.0.0.0 et en mettant à la place seulement l'adresse ip publique, fournie par AWS Fargate.



The screenshot shows the 'Inbound rules' tab of a security group configuration. It displays one rule: 'Inbound rules (1)'. The rule details are as follows:

Security group rule ID	IP version	Type	Protocol	Port range	Source
sgr-0c2a64feba79654e1	IPv4	HTTP	TCP	80	0.0.0.0/0

At the top right of the table, there are buttons for 'Manage tags' and 'Edit inbound rules'. Navigation arrows and a search bar are also visible.

Cliquer sur Run task

Amazon Elastic Container Service > Clusters

Clusters (1)

Search clusters

By default, we only load up to 1,000 clusters at a time.

Cluster	Services	Tasks	Container instances
wordpress-fargate-cluster	0	0 Pen... 1 Run...	0 EC2

On voit que la tâche est lancée

5 — Accéder à WordPress

5.1 Cliquer sur la task en cours

Onglet Networking

Copier l'Public IP : dans notre cas, par exemple , l'IP publique fournie automatiquement par AWS Fargate est : 3.98.120.71

Amazon Elastic Container Service > Clusters > wordpress-fargate-cluster > Tasks > 922904382c864018917ad2ada241a0a8 > Configuration

Encryption

Default AWS Fargate encryption

Size (GiB)

20

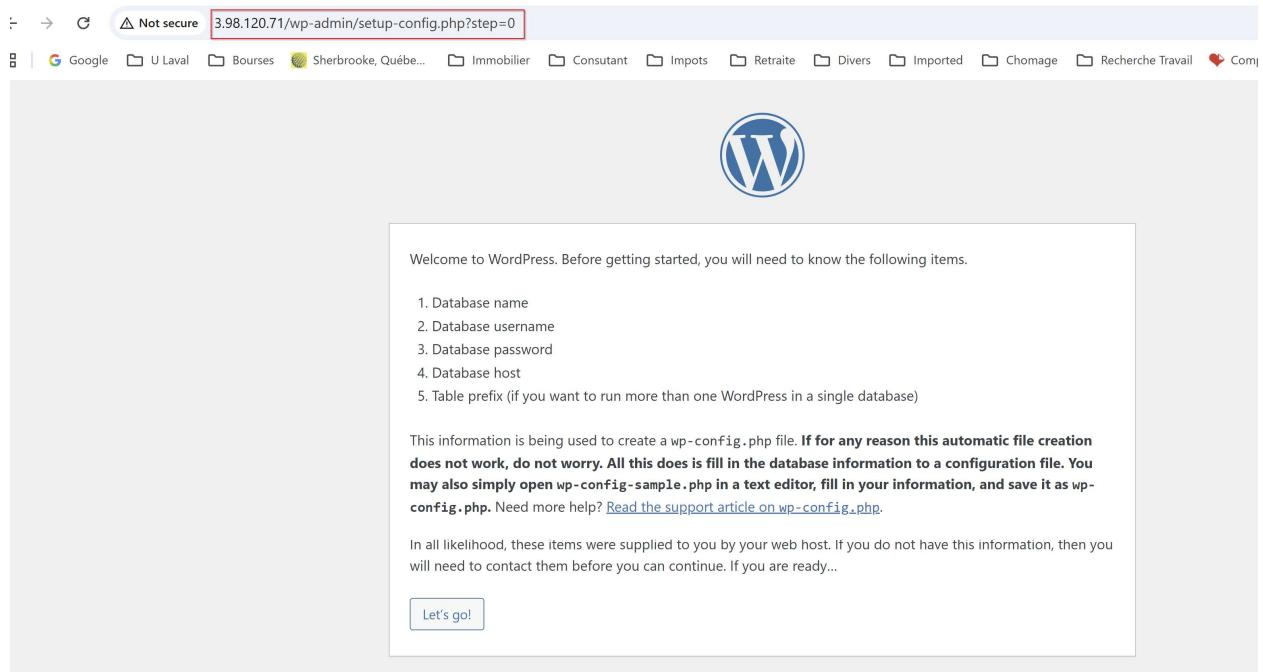
Configuration

Operating system/Architecture Linux/X86_64	Capacity provider Fargate	ENI ID eni-05a3ffbec319e23d4
CPU Memory .5 vCPU 1 GiB	Launch type Fargate	Network mode awsvpc
Platform version 1.4.0	Task definition: revision wordpress-task1	Subnet subnet-07b548eec9ff23574
Task execution role ecsTaskExecutionRole	Task group family:wordpress-task	Private IP 172.31.8.56

5.2 Tester dans le navigateur Windows

Dans le navigateur : http://PUBLIC_IP

On voit la page d'installation WordPress.



Ps : si le subnet est public, l'Adresse ip publique et si le groupe de sécurité permet en inbound le trafic http (protocole TCP et numéro de port 80) , alors la connexion sera fonctionnelle

On peut aussi tester la connectivité avec l'outil : curl

```
C:\Users\henri>curl http://3.98.120.71:80
```

```
C:\Users\henri>
```

Dans le cas où le groupe de sécurité ne seraient pas configuré correctement, on aurait le type d'erreur suivant, par exemple :

```
© MICROSOFT CORPORATION. All rights reserved.  
C:\Users\henri>curl http://3.98.120.71:80  
curl: (28) Failed to connect to 3.98.120.71 port 80 after 21049 ms: Could not connect to server
```

6 — Vérifications

6.1 Vérifications CLI

- ◆ Lister les clusters ECS :

```
aws ecs list-clusters
```

Retour attendu :

arn:aws:ecs:ca-central-1:123456789012:cluster/wordpress-fargate-cluster

```
C:\Users\henri>
C:\Users\henri>aws ecs list-clusters
{
    "clusterArns": [
        "arn:aws:ecs:ca-central-1::cluster/wordpress-fargate-cluster"
    ]
}
C:\Users\henri>
```

- ◆ Lister les tasks d'un cluster :

```
aws ecs list-tasks --cluster wordpress-fargate-cluster
```

👉 Retourne les ARN des tasks en cours

```
C:\Users\henri>aws ecs list-tasks --cluster wordpress-fargate-cluster
{
    "taskArns": [
        "arn:aws:ecs:ca-central-1::task/wordpress-fargate-cluster/922904382c864018917ad2ada241a0a8"
    ]
}
C:\Users\henri>
```

- ◆ Décrire les tasks (containers, IP, status) :

```
aws ecs describe-tasks \
--cluster wordpress-fargate-cluster \
--tasks <TASK_ARN>
```

- ◆ Lister les task definitions :

```
aws ecs list-task-definitions
```

-
- ◆ Décrire la task definition :

```
aws ecs describe-task-definition \
--task-definition wordpress-task
```

👉 Utile pour vérifier CPU, mémoire, ports, image

6.2 vérifications via la console AWS Fargate

- ◆ Voir le cluster

AWS Console → ECS → Clusters

- ◆ Voir les tasks en cours

ECS → Clusters → wordpress-fargate-cluster → Tasks

- ◆ Voir le container WordPress

Cliquer sur la task

Onglet Containers