Break a Monolith Application into Microservices

Overview

In this tutorial, you will deploy a monolithic node.js application to a Docker container, then decouple the application into microservices without any downtime. The node.js application hosts a simple message board with threads and messages between users.

Module 1 Containerize the Monolith

Step 1 Get Setup

- 1. Have AWS account
- 2. Install Docker
- 3. Install the AWS CLI

after using pip install awscli —upgrade —user, use aws —version to verify. Shows aws command not found. Try another:

```
[YimatoMacBook-Pro:api yiqian$ aws
-bash: aws: command not found
[YimatoMacBook-Pro:api yiqian$ ~/.local/bin/aws --version
aws-cli/1.16.43 Python/3.6.3 Darwin/18.0.0 botocore/1.12.33
YimatoMacBook-Pro:api yiqian$
```

4. Have a Text Editor

Step 2 Download & Open the Project

Download from GitHub

Step 3 Provision a Repository

The repository address

```
请确保您已安装 AWS CLI 和 Docker 的最新版本。了解更多信息,请参阅 ECR 文档。

1) 检索登录命令,对 Docker 客户端进行身份验证,以允许其访问您的注册表:
对于 macOS 或 Linux 系统,请使用 AWS CLI:

S(aws ecr get-login --no-include-email --region us-east-1)

对于 Windows 系统,请使用适用于 PowerShell 的 AWS 工具:

Invoke-Expression -Command (Get-ECRLoginCommand -Region us-east-1).Command

注意: 如果您在使用 AWS CLI 时收到 "Unknown options: --no-include-email" 错误,请确保您已安装了最新版本。了解更多

2) 如果您使用的是 AWS CLI, 请从步骤 1 的输出运行登录命令。
3) 使用以下命令生成 Docker 映像。有关从头开始生成 Docker 文件的信息,请参阅此处的说明。如果您已生成映像,则可跳过此步骤:

docker build -t api 。

4) 生成完成后,标记您的映像,以便将映像推送到此存储库:

docker tag api:latest 747142172942.dkr.ecr.us-east-1.amazonaws.com/api:latest

5) 运行以下命令将此映像推送到您新创建的 AWS 存储库:

docker push 747142172942.dkr.ecr.us-east-1.amazonaws.com/api:latest
```

Step 4 Build & Push the Docker Image

1. Configure your credentials

```
YimatoMacBook-Pro:api yiqian$ ~/.local/bin/aws configure

AWS Access Key ID [None]: AKIAJZITGLEDT7MLXEBQ

AWS Secret Access Key [None]: DEciIllb+bA38MEB2W92WeChDc0WdiLbJVHqlQIm

Default region name [None]: us-east-2

Default output format [None]: json

YimatoMacBook-Pro:api yiqian$

[YimatoMacBook-Pro:api yiqian$ ~/.local/bin/aws ecr get-login --no-include-email --region us-west-2

An error occurred (UnrecognizedClientException) when calling the GetAuthorizatio nToken operation: The security token included in the request is invalid.

YimatoMacBook-Pro:api yiqian$ ~/.local/bin/aws ecr get-login --no-include-email --region us-west-2
```

Add json permission in User

```
東蛤燗妛
        {}JOUN
                    猵뙾朿蚧
 1 - {
        "Version": "2012-10-17",
        "Statement": [
 3 +
 4.-
             {
                 "Effect": "Allow",
                 "Action": [
 6 -
                     "ecr:GetAuthorizationToken",
                     "ecr:InitiateLayerUpload",
                     "ecr:UploadLayerPart",
                     "ecr:CompleteLayerUpload",
                     "ecr:BatchCheckLayerAvailability",
                     "ecr:PutImage"
                 ],
                 "Resource": "*"
```

```
Default region name [us-east-1]:
Default output format [json]:
YimatoMacBook-Pro:api yiqian$ ~/.local/bin/aws ecr get-login --no-include-email
   -region us-east-1
docker login -u AWS -p eyJwYXlsb2FkIjoibEEzWnhHbm8zczgyOFNpcFMwT3p1anJRQnBwWHpCN
WM4aHY3K3I2NGVSRENQTGF5U01yeTZWRnZaajJwVVpSb3RqMEJ2OWM3QnkyVUZXWEsySys1ZWUwVHkva
XVQWlPa09YQzJsRmtLWDRRRXNtQXI0b1FqelhuNmFMWnkrZ0RyME9xSEpTYUVFUmdpdErtMVNCVFBIN
WR1eTA3ei9NV2hvRCtLUklqb1MwRnE2UHEwRU5XN2pHQkZPKzM4VWtlQkFwMlBrbWc2czllZXNaRG9yT
GFLbENZbWZVME9tdUpSQzJOdHNBbGNPbkxiRG1UbXhNMDJ2ZkthSmpwWnZ6V1RHRjVJdFgzWGR3c0JYe
EFKeEx2UG54R2dwbzUxSW54L0RCT0hpcFZGTlpFSnRQQjh0WUR5Y1hsK0VHNVA1NUVv0WxJczlKWWEvM
3NiL29zaWFyWHk2RkZnNkF1VHg4ckpz0UpHN01nZGxCUUF2dHREeGN0cklJYVFadEZtWldqRmZWMzZyR
HdIZEZpbmxVV0pMOStSUDVnYzdWeGZBOW9ZSTVUY3RaQVhRNG9TSTRvVXR0bUVHRms4OFBZZG8wL1Q5V
{\tt GJnYXJYbUNHQTNUZDRuQ1VuakpGVm1YdkoyW1h3SWRubjFRdUNPKzIxbmwxbz16aHEwK0dxSXZIT2IrValue} \\
3U0V1hCeldinknQY0dyVVZKZG0rb2EwRVhQWWdKbCsyS1VLWCtJb3R0TFMrOHVBdUZnZEVNQVpWMWtvM
\tt 0doOURtZk1wcm91by9TWj1rRktxSEFUT3M2RjBXZHJCRHpPd0hQTDZzTmVzS1J1eEFKaTNwbVdCQy9Carreless and the temperature of the temperat
U52QUVVeG53T2JEN3hKMUhmVXkweHlHaDdYTjZMek1SZlIzVzJrd0h2THZVdzJBRmtjUzF2dUk4Mm4yY
 ThKamo0MzI2eGJZUEUvWUJ4cFAwVTRDS0Nqb3NuemZMOUJ6ZnpOYWJQeVd6K3A3NG1ZR1g2M1NPRWpjM
mc1Uzg0U0NWN0J3a0RoSkFBd0JPRGpSYVVqc3UyaUt2MnZROE9I0EJtNi9GdmNqb1c1Y0xNWVJOK3I4R
WdXZkdSZ3FTbWNBbjZ6TytVVld00HVGKzBvclJXM3VUd2JtbW1vQndnNEVMcTc0Rm9VK0lqUjZyY3QvK
1BmTFRDMXpBVStwNkRSNHNiNFdON2p1U1NlaTBDM0tzTUV4dDdmcWJ6eEkyL21MSmM0ZDVCVjVBN3M3T
HFsek1NOD12SXdkcDZ0WFVMNEVWOGEya0JQb11uWmwrVD1QSUZSK3RuZW5kM0hYeU5hM3RKbnFYSXBXY
3hrOVVPQVZpS3Fvb25qekRFb2tQZndSR0djS3NNeitRWER5TEVYW1JjdmRLWjEzcGV5cX1WcTR1MjBaZ
lcxU29hcFRlME81MkFKT0srOXhnYkVnPT0iLCJkYXRha2V5IjoiQVFFQkFIaHdtMFlhSVNKZVJ0Sm01b
jFHNnVxZWVrWHVvWFhQZTVVRmN10VJx0C8xNHdBQUFINHdmQvlKS29aSWh2Y05BUWNHb0c4d2JRSUJBR
EJvQmdrcWhraUc5dzBCQndFd0hnWUpZSVpJQVdVREJBRXVNQkVFREZnQnNhRE15cFRnM1F1V1NBSUJFS
{\tt UE3TmozUXd1eUZTQTVEMEJWS2RibyswTDgrWnNIek1RR0N4Rkh5VFMvc3ZLYXZSemE5dHFiK310b3dBd}
nRUa2tlQVpzVjBmRldtQTNyL3Npaz0iLCJ2ZXJzaW9uIjoiMiIsInR5cGUi0iJEQVRBX0tFWSIsImV4c
GlyYXRpb24i0jE1NDA5MDc5MjV9 https://747142172942.dkr.ecr.us-east-1.amazonaws.com
YimatoMacBook-Pro:api yiqian$
```

Build the Image

Push the image to ECR:



Module 2 Deploy the Monolith

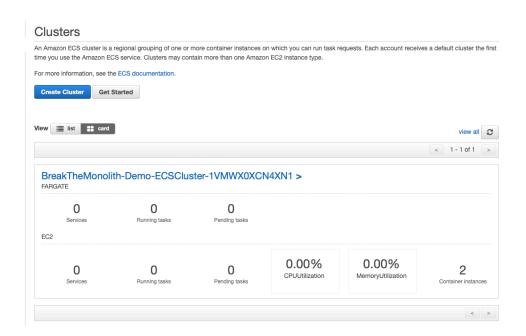
Step 1 Launch an ECS Cluster using AWS CloudFormation

Create stack

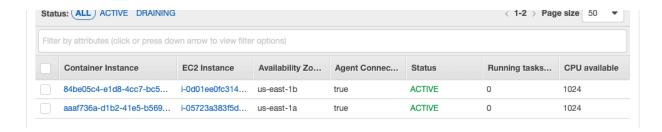


Step 2 Check your Cluster is Running

Cluster



2 EC2 instances



Step 3 Write a Task Definition

Create task definition



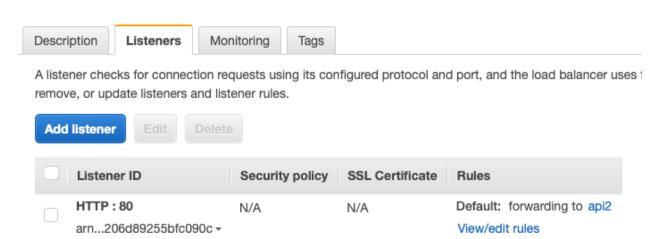
Step 4 Configure the Application Load Balancer: Target Group

Configure the ALB Target Group



Step 5 Configure the Application Load Balancer: Listener

Add listener



Step 6 Deploy the Monolith as a Service

Create Service

| Servic | e : ap | i | | | | | | |
|------------------|-------------|---|----------------|-------------|---------|------|---------------|---|
| | Cluster | | | | | ı | Desired count | 1 |
| BreakTh | neMonolith- | Demo-ECS | Cluster-1VMWX0 | XCN4XN1 | | | | |
| Status | | ACTIVE | | | | | ending count | 0 |
| Task definition | | api:1 | | | | | Running count | 0 |
| Service type | | REPLICA | | | | | | |
| Launch type | | FARGATE | | | | | | |
| Platform version | | LATEST(1.2.0) | | | | | | |
| Service role | | aws-service-role/ecs.amazonaws.com/AWSServiceRoleForECS | | | | | | |
| Details | Tasks | Events | Auto Scaling | Deployments | Metrics | Logs | | |

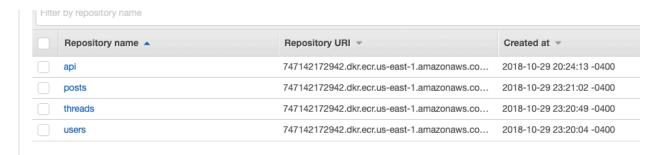
Step 7 Test your Monolith



Part 3 Break the Monolith

Step 1 Provision the ECR Repositories

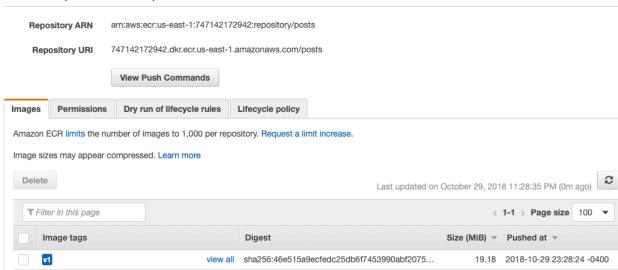
Create repository

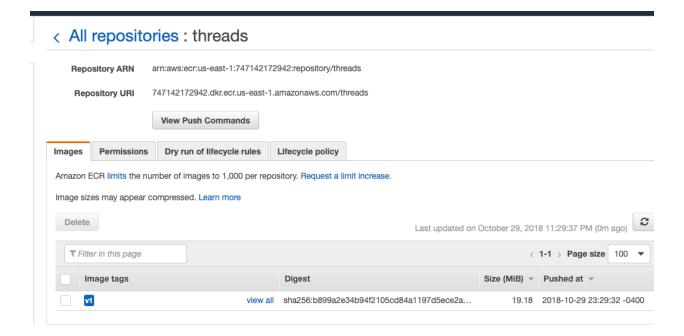


Step 3 Build and Push Images for Each Service

Build images tag then push







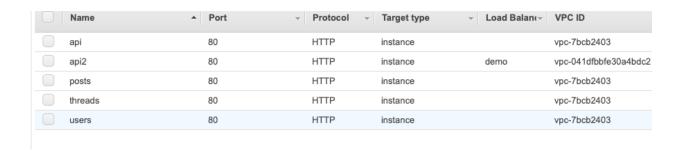
< All repositories : users arn:aws:ecr:us-east-1:747142172942:repository/users Repository ARN 747142172942.dkr.ecr.us-east-1.amazonaws.com/users Repository URI View Push Commands Dry run of lifecycle rules Images Amazon ECR limits the number of images to 1,000 per repository. Request a limit increase. Image sizes may appear compressed. Learn more Delete Last updated on October 29, 2018 11:30:25 PM (0m ago) ▼ Filter in this page < 1-1 > Page size 100 ▼ Image tags Digest Size (MiB) - Pushed at sha256:54b052f169a3975cbf8669734671f4cdd3... 19.18 2018-10-29 23:30:25 -0400 v1

Part 4 Deploy Microservices

Step 1 Write Task Definition for your Services

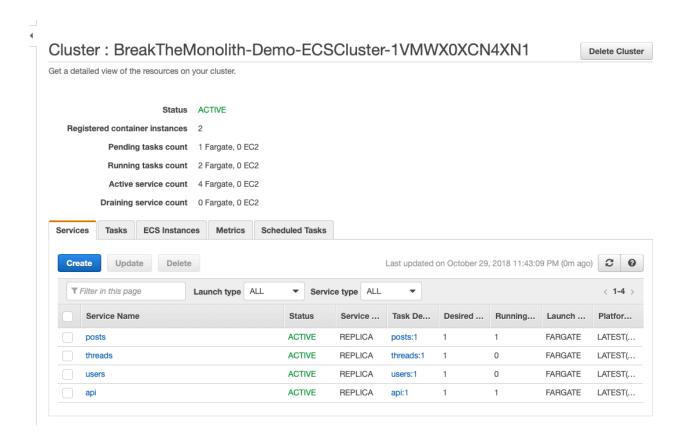


Step 2 Configure the Application Load Balancer: Target Groups



Step 3 Configure Listener Rules

Step 4 Create Services



Step 6 Validate your Deployment

{"id":3,"title":"In search of a new guitar","createdBy":1}