Pre-Requirements:

- Ec2 Instance
- Docker
- Aws-cli

Step1: Connect to the server using the cmd or gitbash:

• Ssh -i "fcra.pem" ubuntu@ip

Step2: Installation of Docker:

- apt-get install docker.io
- systemctl start docker
- systemctl enable docker
- systemctl status docker

•

Step2: Installation of Aws Cli:

- \bullet curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o
- "awscliv2.zip"
- unzip awscliv2.zip
- apt install unzip -y
- unzip awscliv2.zip
- sudo ./aws/install (To check the path where it installed)
- AWS –version
- AWS configure (enter the id and password details of the iam role which uhave been created)

Step3:

Create a role in AWS with the access of ec2 repository and admin access and create access key:

- for further requirements example below. * *
- Access key XXXXXXXXXXXXXXX
- Secret access key: XXXXXXXXXXXXXXX
- Region : ap-south-1

Step4:

Once you completed the above steps use the Commands to generate the self-signed certificate:

- * openssl req -newkey rsa:4096 -x509 -sha256 -days 3650 -nodes -out protiviti.crt -keyout protiviti.key
- * openssl pkcs12 -export -out protiviti.pfx -inkey protiviti.key -in protiviti.crt

Step5:

Then include in the certificate path in docker example below.

```
FROM mcr.microsoft.com/dotnet/aspnet:6.0-alpine AS base
WORKDIR /app
EXPOSE 80
EXPOSE 443
RUN apk apk update && apk upgrade &&
 \apk add --no-cache openssl \
 tzdata \
 icu-libs>=67
 krb5-libs>=1.18
 libgcc>=10.3
 libintl>=0.21 \
 libss11.1>=1.1.1 \
 libstdc++>=10.3
 \zlib>=1.2.11
ENV TZ=Asia/Calcutta
ENV
ASPNETCORE_ENVIRONMENT=testing
DOTNET RUNNING IN CONTAINER=tru
ENV DOTNET_SYSTEM_GLOBALIZATION_INVARIANT=false
FROM mcr.microsoft.com/dotnet/sdk:6.0-alpine AS build
WORKDIR /src
COPY ["FCRA.Web/FCRA.Web.csproj", "FCRA.Web/"]
COPY ["FCRA.Common/FCRA.Common.csproj", "FCRA.Common/"]
COPY ["FCRA.Repository/FCRA.Repository.csproj",
"FCRA.Repository/"]COPY ["FCRA.Models/FCRA.Models.csproj",
"FCRA.Models/"]
COPY ["FCRA.ViewModels/FCRA.ViewModels.csproj",
"FCRA.ViewModels/"]RUN dotnet restore "FCRA.Web/FCRA.Web.csproj"
COPY..
WORKDIR "/src/FCRA.Web"
RUN dotnet build "FCRA.Web.csproj" -c Release -o
/app/buildFROM build AS publish
```

```
RUN dotnet publish "FCRA.Web.csproj" -c Release -o /app/publish /p:UseAppHost=false //ARG CERT_PASSWORD //RUN dotnet dev-certs https -ep /app/ewraprotiviti.pfx -p ewraprotiviti@123

FROM base AS final WORKDIR /app COPY --from=publish /app/publish . //RUN mkdir -p /https //COPY ewraprotiviti.pfx /app COPY ewraprotiviti.pfx /https ENTRYPOINT ["dotnet", "FCRA.Web.dll"] * docker build -t <nameof_dockerfile> . * docker build -t abcd .
```

Step6:

Then run the application with environment variables in docker using the command below.

```
docker run -d -p 80:80 -p 443:443 \
-e ASPNETCORE_URLS="https://+;http://+" \
-e ASPNETCORE_HTTPS_PORT=443 \
-e ASPNETCORE_Kestrel__Certificates__Default__Password="protiviti@123" \
-e ASPNETCORE Kestrel Certificates Default Path="/https/protiviti.pfx" \
-e MetadataAddress="https://login.microsoftonline.com/ba04dd9d-19c9-423e-85c1-
63bc63f9ff4c/federationmetadata/2007-06/federationmetadata.xml?appid=d58bfe58-9cf7-4e27-84ec-
39fd728ab156" \
-e username="sa" \setminus
-e password="fcra@123" \
-e host="13.201.123.96" \
-e port=""\
-e bucketname=""\
-e AWSRegion="ap-south-1" \
 -e dbInstanceIdentifier="RISKDBADCB" \
 -e engine=""\
-e IsThroughSMTP="N" \
-e From=""\
-e SMTPUsername=""\
-e SMTPPassword=""\
-e SMTPHost=""\
-e SMTPPort="587" \
-e RealmUrl="https://ec2-3-6-40-111.ap-south-1.compute.amazonaws.com/" \
 -e certificatepath=""\
 -e certificatepassword=""\
 -e httpport=""∖
-e httpsport=""\
-e IsSSOApplicable="Y" \
 -e IsEnvironmentVariableApplicable="Y"
\Abcd
```

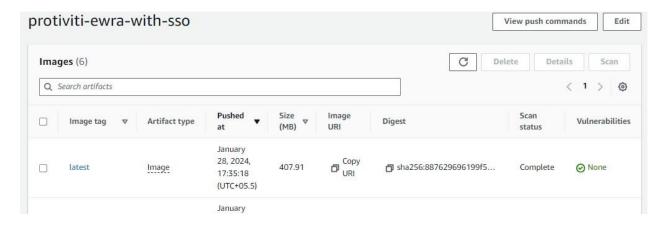
```
\times
🕞 Y 💂
                                                                   x \[ +
           CYSPC
FCRA.Web/FCRA.Web.csproj", "FCRA.Web/"]
FCRA.Common/FCRA.Common.csproj", "FCRA.Common/"]
FCRA.Repository/FCRA.Repository.csproj", "FCRA.Repository/"]
FCRA.Models/FCRA.Models.csproj", "FCRA.Models/"]
FCRA.ViewModels/FCRA.ViewModels.csproj", "FCRA.ViewModels/"]
ret restore "FCRA.Web/FCRA.Web.csproj"
 .
KKDIR "/src/FCRA.Web"
N dotnet build "FCRA.Web.csproj" -c Release -o /app/build
ROM build AS publish
UN dotnet publish "FCRA.Web.csproj" -c Release -o /app/publish /p:UseAppHost-false
DPY --from=publish /app/publish .
NN mkdir -p /https
```

Step7:

Push the image into ECR with the commands below.

* aws ecr get-login-password --region ap-south-1 | docker login --username AWS --password-stdin 240887461522.dkr.ecr.ap-south-1.amazonaws.com

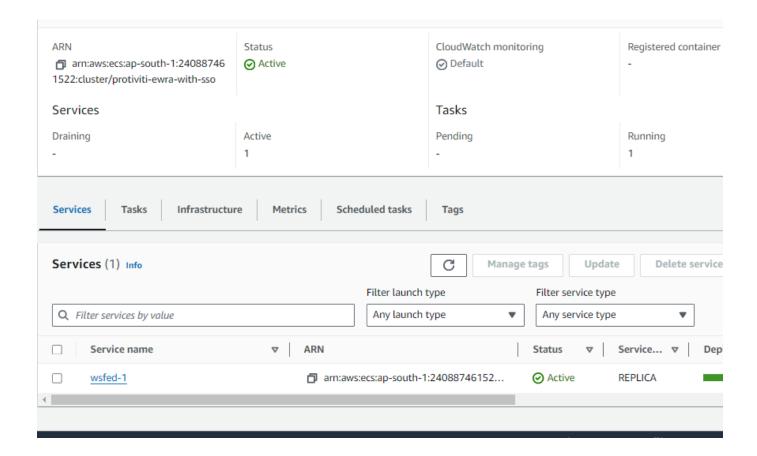
- * docker tag abcd:latest 240887461522.dkr.ecr.ap-south-1.amazonaws.com/protiviti-ewra-with-sso:latest
- * docker push 240887461522.dkr.ecr.ap-south-1.amazonaws.com/protiviti-ewra-with-sso:latest



Step8:

Then Create a task definition in AWS ECS and attach the environment variables with the values.

• Create a task definition with the below configurations.



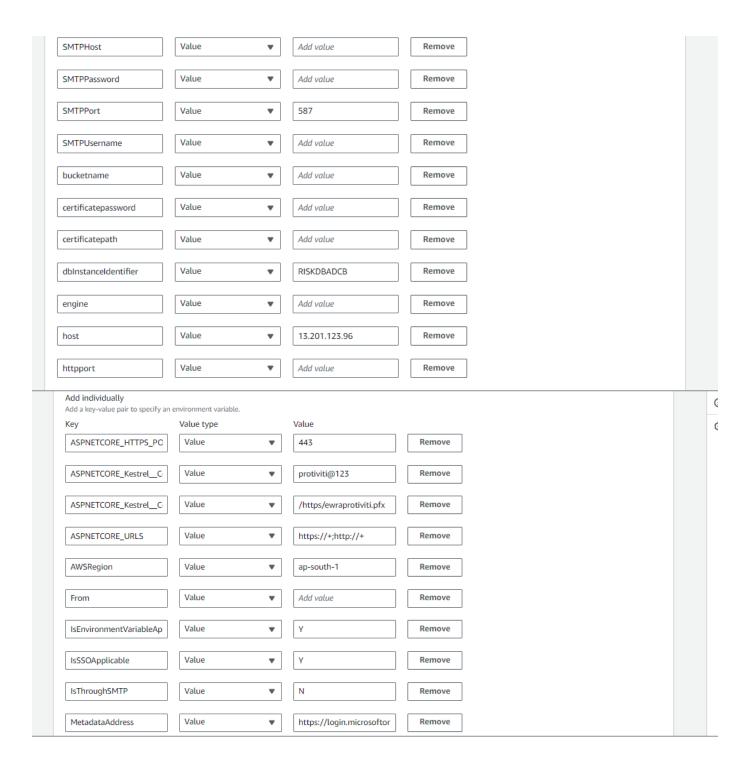
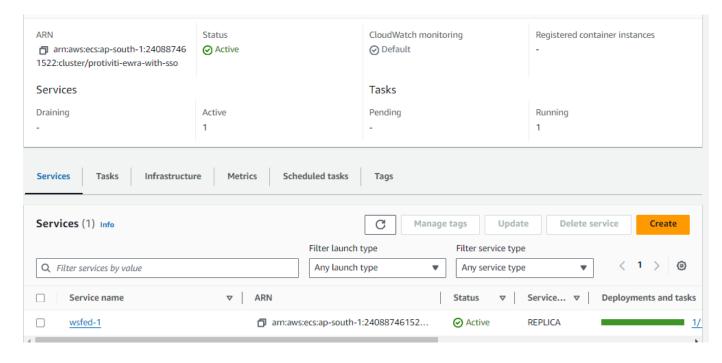


Image URI 240887461522.dkr.ecr.ap-south-1.amaze en use the credentials to reference images in private re access ports on the host to send or receive traffic. For ol Port name wrat-80-tcp	egistries. r port name, a default will be assigned if left b App protocol	Essential container Yes	•
en use the credentials to reference images in private re access ports on the host to send or receive traffic. For port_name	egistries. r port name, a default will be assigned if left b App protocol		•
access ports on the host to send or receive traffic. For	r port name, a default will be assigned if left b	lank.	
ol Port name	App protocol	ank.	
▼ ewrat-80-tcp			
	HTTP ▼	Remove	
▼ ewrat-443-tcp	App protocol ▼	Remove	
ner is given read-only access to its root file system.			
Info s are different from task-level values. They define how nated.	w much resources are allocated for the contain	er. If container attempts to exceed th	ne memory
	Source revision		
	20		
ts for the task definition.			
c definition parameters. ton EC2 instances.			
		:2 instances.	:2 instances.

Step9: Then attach the created task for the Cluster shown in the below image.



```
"name": "AWSRegion",
39
                         "value": "ap-south-1"
40
                     },
41
                         "name": "SMTPPort",
42
43
                         "value": "587"
44
                     },
45
                         "name": "password",
46
47
                         "value": "fcra@123"
48
49
50
                         "name": "SMTPPassword",
                         "value": ""
51
52
53
54
                         "name": "engine",
55
                         "value": ""
56
57
                         "name": "bucketname",
58
59
                         "value": ""
60
61
62
                         "name": "httpport",
                         "value": ""
63
64
65
                                                                                                                                    JSON
           "taskDefinitionArn": "arn:aws:ecs:ap-south-1:240887461522:task-definition/protiviti-ewra-with-sso:12",
```

```
"containerDefinitions": [
4
                "name": "protiviti-ewra-with-sso",
5
               "image": "240887461522.dkr.ecr.ap-south-1.amazonaws.com/protiviti-ewra-with-sso:latest",
6
               "cpu": 0,
               "portMappings": [
8
9
                  {
10
                       "name": "protiviti-ewra-with-sso-80-tcp",
11
                       "containerPort": 80,
12
                       "hostPort": 80,
                       "protocol": "tcp",
13
14
                       "appProtocol": "http"
15
16
17
                       "name": "protiviti-ewra-with-sso-443-tcp",
18
                       "containerPort": 443,
                       "hostPort": 443,
19
                       "protocol": "tcp"
20
21
22
               ],
                "essential": true,
23
24
               "environment": [
25
                  {
26
                       "name": "IsSSOApplicable",
                 "value": "Y"
```

```
"name": "ecs.capability.task-eni"
176
177
178
179
                  "name": "com.amazonaws.ecs.capability.docker-remote-api.1.29"
180
181
182
         "placementConstraints": [],
         "compatibilities": [
183
184
             "EC2",
185
             "FARGATE"
186
         "requiresCompatibilities": [
187
188
             "FARGATE"
189
         ],
190
         "cpu": "1024",
         "memory": "3072",
191
192
         "runtimePlatform": {
193
             "cpuArchitecture": "X86_64",
             "operatingSystemFamily": "LINUX"
194
195
         "registeredAt": "2024-01-28T12:08:07.696Z",
196
         "registeredBy": "arn:aws:iam::240887461522:user/sachchidanandjha",
197
         "tags": []
198
199 }
```

