



School of Computer Science Engineering and Application

BCA TY SEM VI

Subject Name: Container and Orchestration Practical

Assignment No. 7

Aim: Start Docker Swarm Services, Create A Service, Also Show the Working of Manager and Worker Node.

Submitted By:

Name: Jayesh Bhangale

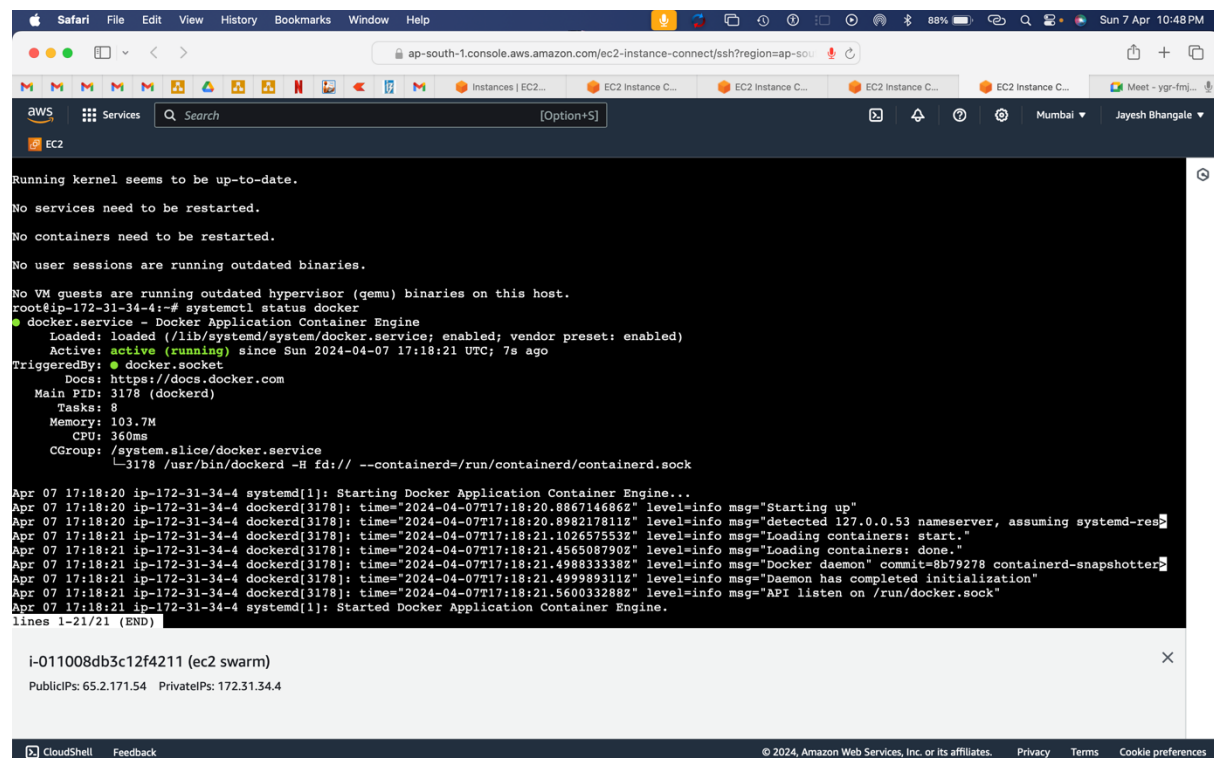
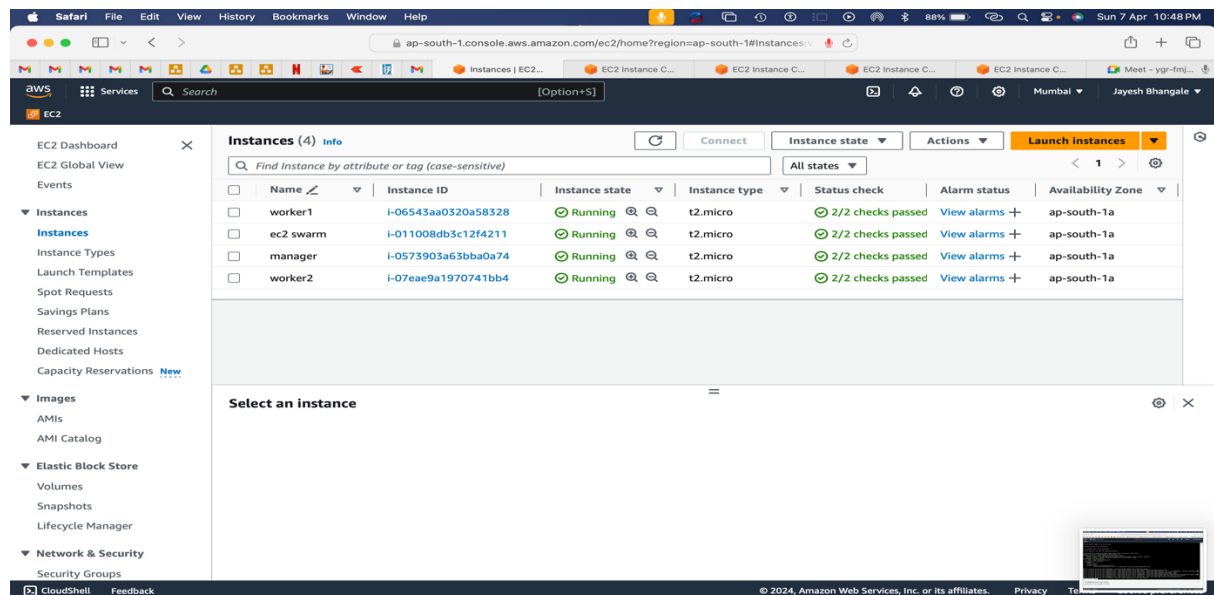
PRN: 20210801024

Date: 8th April, 2024

Aim: Start Docker Swarm Services, Create A Service, Also Show the Working of Manager and Worker Node.

Technology Used: AWS

Step 1: Create An Instance and Install Docker Check If It Is Running.



Step 2: Activate The Docker Swarm

docker swarm init

```

44 updates can be applied immediately.
28 of these updates are standard security updates.
To see these additional updates run: apt list --upgradable

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status

Last login: Sun Apr  7 16:47:07 2024 from 13.233.177.5
ubuntu@ip-172-31-38-196:~$ sudo -i
root@ip-172-31-38-196:~# docker join-token worker
docker: 'join-token' is not a docker command.
See 'docker --help'
root@ip-172-31-38-196:~# docker swarm join-token worker
To add a worker to this swarm, run the following command:

    docker swarm join --token SWMTKN-1-2o4bwoa5wgxnktimo80gzg08b8ippjks8d13lwt5ueg9qdpxc-ajt9v8j4ui8hw67uox0glc78i 172.31.38.196:2377

root@ip-172-31-38-196:~# docker swarm init
Error response from daemon: This node is already part of a swarm. Use "docker swarm leave" to leave this swarm and join another one.
root@ip-172-31-38-196:~# docker swarm leave --force
Node left the swarm.
root@ip-172-31-38-196:~# docker swarm init
Swarm initialized: current node (xlwjqwiesxj4ayjrhs0r2yzcx) is now a manager.

To add a worker to this swarm, run the following command:

    docker swarm join --token SWMTKN-1-6dfjusy645o29krp2gzm8hzl9odbkqt8p0dwkxh9ih5vol17mrlj5 172.31.38.196:2377

To add a manager to this swarm, run 'docker swarm join-token manager' and follow the instructions.

root@ip-172-31-38-196:~#
  
```

i-0573903a63bba0a74 (manager)

PublicIPs: 13.201.122.122 PrivateIPs: 172.31.38.196

Step 3: Create A Service and Ping A Website

Docker service create alpine ping www.google.com

The screenshot shows an AWS CloudShell terminal session on an EC2 instance. The user runs the command `docker service ls`, which displays a table of services. The service `2pzzxxffaltm7` is in a replicated state with 1/1 replicas. The user then runs `docker service ps 2pzzxxffaltm7`, showing the service is running on node `ip-172-31-38-196`. Next, the user runs `docker service create alpine ping www.google.com`, which creates a new service. The user then runs `docker service ls` again, showing the new service `ii7gihbsyqjjx5aqop9p2dlt8` in a replicated state. Finally, the user runs `docker service scale 2pzzxxffaltm7=5`, scaling the service to 5 replicas. The terminal output shows the progress of the scaling operation, with 5 out of 5 tasks completed. The user then runs `docker service ls` again, showing the scaled service. The terminal also displays the public and private IP addresses of the instance.

```
--tlskey string      Path to TLS key file (default "/root/.docker/key.pem")
--tlsverify          Use TLS and verify the remote
-v, --version        Print version information and quit

Run 'docker COMMAND --help' for more information on a command.

For more help on how to use Docker, head to https://docs.docker.com/go/guides/

root@ip-172-31-38-196:~# docker service ls
ID            NAME            MODE            REPLICAS    IMAGE            PORTS
2pzzxxffaltm7 blissful_colden replicated 1/1          alpine:latest
root@ip-172-31-38-196:~# docker service ps 2pzzxxffaltm7
ID            NAME            IMAGE            NODE            DESIRED STATE    CURRENT STATE    ERROR    PORTS
g2intqtoexkz blissful_colden.1 alpine:latest    ip-172-31-38-196 Running          Running 3 minutes ago
root@ip-172-31-38-196:~# docker service create alpine ping www.google.com
ii7gihbsyqjjx5aqop9p2dlt8
overall progress: 1 out of 1 tasks
1/1: running [=====>]
verify: Service ii7gihbsyqjjx5aqop9p2dlt8 converged
root@ip-172-31-38-196:~# docker service ls
ID            NAME            MODE            REPLICAS    IMAGE            PORTS
2pzzxxffaltm7 blissful_colden replicated 1/1          alpine:latest
ii7gihbsyqjj nostalgic_wiles replicated 1/1          alpine:latest
root@ip-172-31-38-196:~# docker service scale 2pzzxxffaltm7=5
2pzzxxffaltm7 scaled to 5
overall progress: 5 out of 5 tasks
1/5: running [=====>]
2/5: running [=====>]
3/5: running [=====>]
4/5: running [=====>]
5/5: running [=====>]
verify: Service 2pzzxxffaltm7 converged
root@ip-172-31-38-196:~#
```

i-0573903a63bba0a74 (manager)
PublicIPs: 13.201.122.122 PrivateIPs: 172.31.38.196

The screenshot shows an AWS CloudShell terminal session on an Ubuntu instance. The user runs the command `sudo docker service create alpine ping www.google.com`, which creates a new service. The user then runs `docker service ls`, showing the service `n9z84f97gu50yfwe27s9xtiz` in a replicated state. The user then runs `docker service ps n9z84f97gu50yfwe27s9xtiz`, showing the service is running on node `ip-172-31-36-77`. Finally, the user runs `docker service scale n9z84f97gu50yfwe27s9xtiz=5`, scaling the service to 5 replicas. The terminal output shows the progress of the scaling operation, with 5 out of 5 tasks completed. The user then runs `docker service ls` again, showing the scaled service. The terminal also displays the public and private IP addresses of the instance.

```
apparmor
seccomp
Profile: builtin
Kernel Version: 5.15.0-1055-aws
Operating System: Ubuntu 20.04.6 LTS
OSType: linux
Architecture: x86_64
CPUs: 1
Total Memory: 954.1MiB
Name: ip-172-31-36-77
ID: 15ad0229-59da-4e4a-8881-3f7624fad3d
Docker Root Dir: /var/lib/docker
Debug Mode: false
Experimental: false
Insecure Registries:
127.0.0.0/8
Live Restore Enabled: false

ubuntu@ip-172-31-36-77:~$ sudo docker service create alpine ping www.google.com
n9z84f97gu50yfwe27s9xtiz
overall progress: 1 out of 1 tasks
1/1: running [=====>]
verify: Service n9z84f97gu50yfwe27s9xtiz converged
ubuntu@ip-172-31-36-77:~$ docker service ls
ID            NAME            MODE            REPLICAS    IMAGE            PORTS
n9z84f97gu50 intelligent_ritchie replicated 1/1          alpine:latest
ubuntu@ip-172-31-36-77:~$ sudo docker service ps n9z84f97gu50
ID            NAME            IMAGE            NODE            DESIRED STATE    CURRENT STATE    ERROR    PORTS
qelfutbbj4z intelligent_ritchie.1 alpine:latest    ip-172-31-36-77 Running          Running 5 minutes ago
ubuntu@ip-172-31-36-77:~$
```

i-Ob68b06801f2efa1a (manager)
PublicIPs: 13.127.181.31 PrivateIPs: 172.31.36.77

Step 4 Scale the Replicated Service

docker service scale id=5

The screenshot shows a terminal window in AWS CloudShell. The user is running Docker commands to start and manage services. The output shows that the service '2pzzxxffaltm7' is running and scaled to 5 tasks. The user also runs 'docker service ls' and 'docker container ls -a' to view the services and containers respectively.

ID	NAME	IMAGE	MODE	REPLICAS	IMAGE	PORTS
2pzzxxffaltm7	blissful_colden.1	alpine:latest	replicated	1/1	alpine:latest	
ii7gihbsyqjj	nostalgic_wiles	alpine:latest	replicated	1/1	alpine:latest	

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
13b5f78f0785	alpine:latest	"ping www.google.com"	27 seconds ago	Up 25 seconds		blissful_colden.2.xeelfiowzyq7iqim7uw06ezq
el2651ecd38f	alpine:latest	"ping www.google.com"	27 seconds ago	Up 25 seconds		blissful_colden.3.0q5ap7ss5p0iq2v7g2vb5je75
64970d638464	alpine:latest	"ping www.google.com"	27 seconds ago	Up 25 seconds		blissful_colden.4.xi9du08yg8rpbw4d0spougjg
eo79a2524723	alpine:latest	"ping www.google.com"	27 seconds ago	Up 25 seconds		blissful_colden.5.adx7wminixae7lpezay8vz7pk
6139a38c1336	alpine:latest	"ping www.google.com"	2 minutes ago	Up 2 minutes		nostalgic_wiles.1.px2929h88zifae8xopp2h7n3
4bf47ebda6bd	alpine:latest	"ping www.google.com"	8 minutes ago	Up 8 minutes		blissful_colden.1.g2intqtoexkzqiftn6uol5b2i

i-0573903a63bba0a74 (manager)
PublicIPs: 13.201.122.122 PrivateIPs: 172.31.38.196

Step 5: Create Two New Instance

Instance Name : Worker1 and Worker 2

The screenshot shows the AWS Management Console for EC2 instances. The 'Instances' tab is selected, showing a list of four instances: 'worker1', 'ec2 swarm', 'manager', and 'worker2'. All instances are in the 'Running' state. A modal titled 'Select an instance' is open, showing a search bar and a list of instances to select from.

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone
worker1	i-06543aa0320a58328	Running	t2.micro	2/2 checks passed	View alarms +	ap-south-1a
ec2 swarm	i-011008db3c12f4211	Running	t2.micro	2/2 checks passed	View alarms +	ap-south-1a
manager	i-0573903a63bba0a74	Running	t2.micro	2/2 checks passed	View alarms +	ap-south-1a
worker2	i-07eae9a1970741bb4	Running	t2.micro	2/2 checks passed	View alarms +	ap-south-1a

Step 6: Connect The Other Two Nodes to The Manager Node

docker swarm join-token worker

```

S/5: running [=====]
verify: Service 2pzzxxffaltm7 converged
root@ip-172-31-38-196:~# docker service ls -a
unknown shorthand flag: 'a' in -a
See 'docker service ls --help'.
root@ip-172-31-38-196:~# docker container ls -a
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS        PORTS          NAMES
13b5f78f0785   alpine:latest  "ping www.google.com"   27 seconds ago Up 25 seconds  blissful_colden.2.xeelflowzyg7iqimy7uw06ezq
e12651ec38f    alpine:latest  "ping www.google.com"   27 seconds ago Up 25 seconds  blissful_colden.3.0q5ap7sa5p0iq2v7g2v5jje75
6f970d638464   alpine:latest  "ping www.google.com"   27 seconds ago Up 25 seconds  blissful_colden.4.kl9du08yg8rpbkw4d0zpougjg
e079a2524723   alpine:latest  "ping www.google.com"   27 seconds ago Up 25 seconds  blissful_colden.5.adx7wminixaevlpezay8vz7pk
6139a38c1336   alpine:latest  "ping www.google.com"   2 minutes ago  Up 2 minutes  nostalgic_wiles.1.pxz929h88zifsef8xopp2h7n3
4bf47ebda6bd   alpine:latest  "ping www.google.com"   8 minutes ago  Up 8 minutes  blissful_colden.1.g2intqtoexkzqiftn6uol5b2i
root@ip-172-31-38-196:~# docker service ls
ID                NAME                MODE                REPLICAS        IMAGE          PORTS
2pzzxxffaltm7     blissful_colden     replicated          5/5             alpine:latest
ii7gihbysqjj      nostalgic_wiles     replicated          1/1             alpine:latest
root@ip-172-31-38-196:~# docker service rollback 2pzzxxffaltm7
2pzzxxffaltm7
overall progress: rolling back update: 1 out of 1 tasks
1/1: running [=====]
verify: Service 2pzzxxffaltm7 converged
rollback: rollback completed
root@ip-172-31-38-196:~# docker service ls
ID                NAME                MODE                REPLICAS        IMAGE          PORTS
2pzzxxffaltm7     blissful_colden     replicated          1/1             alpine:latest
ii7gihbysqjj      nostalgic_wiles     replicated          1/1             alpine:latest
root@ip-172-31-38-196:~# docker swarm join-token worker
To add a worker to this swarm, run the following command:

    docker swarm join --token SWMTKN-1-6dfjusy645o29krp2gz8hz19odbkqt8p0dwkxh9ih5voll7mrlj5 172.31.38.196:2377

root@ip-172-31-38-196:~#

```

i-0573903a63bba0a74 (manager)

PublicIPs: 13.201.122.122 PrivateIPs: 172.31.38.196

```

Network: bridge host ipvlan macvlan null overlay
Log: awslogs fluentd gcplogs gelf journald json-file local splunk syslog
Swarm: inactive
Runtimes: io.containerd.runc.v2 runc
Default Runtime: runc
Init Binary: docker-init
containerd version: ae07eda36dd25f8a1b98dfbf587313b99c0190bb
runc version: v1.1.12-0-g51d5e94
init version: de40ad0
Security Options:
  apparmor
  seccomp
   Profile: builtin
  cgroupns
Kernel Version: 6.5.0-1014-aws
Operating System: Ubuntu 22.04.4 LTS
OSType: linux
Architecture: x86_64
CPUs: 1
Total Memory: 949.2MiB
Name: ip-172-31-39-139
ID: 62789543-f342-45c4-9526-5d3169bf2003
Docker Root Dir: /var/lib/docker
Debug Mode: false
Experimental: false
Insecure Registries:
  127.0.0.0/8
Live Restore Enabled: false

root@ip-172-31-39-139:~# docker swarm join --token SWMTKN-1-19j52xs9nmwns3gxmd67q59gtsufkzh32dggld4seta6m7cmt4-f5l5xocgb5a82e6ef5g9lyny1 172.31.36.74:2377
This node joined a swarm as a worker.
root@ip-172-31-39-139:~#

```

i-05ddb89d018586c53 (worker 1)

PublicIPs: 13.201.54.95 PrivateIPs: 172.31.39.139

```

Volume: local
Network: bridge host ipvlan macvlan null overlay
Log: awslogs fluentd gcplogs gelf journald json-file local splunk syslog
Swarm: inactive
Runtimes: io.containerd.runc.v2 runc
Default Runtime: runc
Init Binary: docker-init
containerd version: ae07eda36dd25f8a1b98dfbf587313b99c0190bb
runc version: v1.1.12-0-g51d5e94
init version: de40ad0
Security Options:
  apparmor
  seccomp
   Profile: builtin
  cgroupv2
Kernel Version: 6.5.0-1014-aws
Operating System: Ubuntu 22.04.4 LTS
OSType: linux
Architecture: x86_64
CPUs: 1
Total Memory: 949.2MiB
Name: ip-172-31-44-2
ID: a36e06ae-a779-407c-a12c-cb3fc3172cf1
Docker Root Dir: /var/lib/docker
Debug Mode: false
Experimental: false
Insecure Registries:
  127.0.0.0/8
Live Restore Enabled: false

root@ip-172-31-44-2:~# docker swarm join --token SWMTKN-1-19j52xs9nmwns3gxdm67q59gtsufkzh32dgqld4seta6m7cmt4-f515xocgb5a82e6ef5g9lyny1 172.31.36.74:2377
This node joined a swarm as a worker.
root@ip-172-31-44-2:~#

i-0be3d379f5a32ee11 (worker 2)
PublicIPs: 3.110.143.65 PrivateIPs: 172.31.44.2

```

Step 7: Reflect The Connected Worker Nodes Working Under Manager Node

docker node ls

```

OSType: linux
Architecture: x86_64
CPUs: 1
Total Memory: 949.2MiB
Name: ip-172-31-36-74
ID: f1e1c1a9-7b35-4079-bf28-a0a60072fba0
Docker Root Dir: /var/lib/docker
Debug Mode: false
Experimental: false
Insecure Registries:
  127.0.0.0/8
Live Restore Enabled: false

root@ip-172-31-36-74:~# docker swarm init
Swarm initialized: current node (txldlbn85sejdhvgvt21lbbf5) is now a manager.

To add a worker to this swarm, run the following command:

    docker swarm join --token SWMTKN-1-19j52xs9nmwns3gxdm67q59gtsufkzh32dgqld4seta6m7cmt4-f515xocgb5a82e6ef5g9lyny1 172.31.36.74:2377

To add a manager to this swarm, run 'docker swarm join-token manager' and follow the instructions.

root@ip-172-31-36-74:~# docker swarm join-token worker
To add a worker to this swarm, run the following command:

    docker swarm join --token SWMTKN-1-19j52xs9nmwns3gxdm67q59gtsufkzh32dgqld4seta6m7cmt4-f515xocgb5a82e6ef5g9lyny1 172.31.36.74:2377

root@ip-172-31-36-74:~# docker node ls
ID                HOSTNAME          STATUS    AVAILABILITY    MANAGER STATUS    ENGINE VERSION
txldlbn85sejdhvgvt21lbbf5 * ip-172-31-36-74   Ready     Active           Leader             26.0.0
kxkieef2oos5erevovsuy3dv ip-172-31-39-139 Ready     Active           -                  26.0.0
arwi37e2ltw4ompxjgo2fent4 ip-172-31-44-2   Ready     Active           -                  26.0.0
root@ip-172-31-36-74:~#

i-0d50f03268ac351fa (manager)
PublicIPs: 13.201.123.41 PrivateIPs: 172.31.36.74

```

END OF PRACTICAL

Sign:

Subject In charge -

(Dr. Swapnil D. Waghmare)