Quick Guide to Docker Swarm on AWS

Introduction:
In the realm of modern software development, deploying and scaling applications are paramount. Docker, a leader in containerization, has transformed these processes. This concise guide walks you through setting up a Docker Swarm cluster on AWS, deploying services, ensuring fault tolerance, and horizontally scaling applications.

Step 1: Launching Instances on AWS Cloud:
- Deploy four Amazon Linux instances, meeting minimum hardware requirements for Docker containers.

Step 2: Installing Docker on Instances:
```
yum install docker
systemctl start docker
systemctl enable docker
```

Step 3: Starting Docker Services:
```
systemctl start docker
```

Step 4: Configuring Inbound Rules on Master Node:
- Establish inbound rules on the master node for seamless communication among Swarm nodes.

Step 5: Pinging from Slave to Master:
- Verify connectivity by pinging from slave to master.

Step 6: Initializing Docker Swarm Cluster (On Master):
```
docker swarm init --advertise-addr <Master_IP>
```

Step 7: Listing Nodes in the Cluster:
```
docker node ls
```

Step 8: Adding Worker Nodes:
- During Swarm initialization, execute the provided command on each slave node.

Joining Worker Nodes (Detailed Steps):
- After Swarm initialization, find the join command on the master:
```
  docker swarm join --token SWMTKN-<token> <master-node-ip>:<port>
```
- Copy this command and paste it on each slave node.

Verify Node Joining:
```
docker node ls
```

Deploying and Managing Services:
- In a multi-tier architecture, deploy services for fault tolerance and scalability.

Creating a Service:
```
docker service create --name webserver httpd
```

Listing Services and Service Tasks:
```
docker service ls
docker service ps <service_name>
```

Scaling Services:
```
docker service scale webserver=5
```

Accessing Services with Load Balancer:
```
docker service create --name webserver --publish 8080:80 httpd
```

Monitoring and Scaling:
- Swarm's manager node monitors and automatically relaunches failed containers.
- Scaling in and out is effortless:
```
  docker service scale webserver=10
```

Conclusion:
This guide provides a swift walkthrough of Docker Swarm setup on AWS, service deployment, fault tolerance, and horizontal scaling. Docker Swarm's robust platform offers an optimal solution for modern application deployment. Follow these steps to confidently build and manage your containerized applications.

Check_out_Detailed_blog:-https://medium.com/@srivastavayushmaan1347/a-comprehensive-guide-to-docker-swarm-deploying-and-scaling-containers-on-aws-12098c29f262