1. How do you deploy Docker containers on AWS?

**Amazon Container Service:**

* Amazon ECS is a fully managed container orchestration service that supports Docker containers.

**Steps to Deploy:**

* **Create an ECS cluster:**
* Go to the ECS dashboard in the AWS Management Console.
* Create a new cluster.
* **Create a Task Definition:**
* Define the Docker container image, CPU, memory, networking and other settings.
* Specify the container port mappings and environment variables.
* **Run the Task or Service:**
* Run the task directly or create a service to ensure the container is always running.
* For services, you can configure load balancing and auto-scaling.
* **Access the Application:**
* Use the public IP or DNS of the load balancer to access your application.

**Amazon EC2:**

* You can manually deploy Docker containers on an EC2 instance.

**Steps to Deploy:**

* **Launch an EC2 instance:**
* Launch an EC2 instance with an Amazon Linux or Ubuntu AMI.
* **Install Docker:**
* SSH into the instance and install Docker:

**sudo yum update -y**

**sudo yum install docker -y**

**sudo service docker start**

* **Run the Docker Container:**
* Pull and run your docker image:

**Sudo docker run -d -p 80:80 your-docker-image**

* **Access the Application:**
* Use the public IP address of the EC2 instance to access the application.

1. What is the role of Amazon ECR (Elastic Container Registry)?

* Amazon Elastic Container Registry is a fully managed Docker Container registry provided by AWS.
* It securely stores, manages, and deploys Docker Container Images, making it easier to run containerized applications on AWS services like ECS, EKS, Fargate, Lambda and more.

**Key Roles & Features of Amazon ECR.**

* **Secure storage for Docker Images:**
* Stores private Docker Container images securely.
* Integrates with AWS Identity and Access Management (IAM) for access control.
* Supports Image scanning.
* **Integrates with AWS Container Services:**
* **Amazon ECS** (Elastic Container Service)
* **Amazon EKS** (Elastic Kubernetes Service)
* **AWS Fargate** (Serverless Containers)
* **AWS Lambda** (For container-based lambda functions)
* **AWS EC2** (Elastic Cloud Compute)
* **Performance:**
* Fast retrieval of images for container deployments.
* **Versioning & Tagging:**
* Supports multiple image versions using tags.
* **Security:**
* Supports AWS IAM roles for access control and Encryption.

1. What is the difference between ECS and EKS in AWS?

|  |  |  |
| --- | --- | --- |
| **Feature** | **Amazon ECS** | **Amazon EKS** |
| **Orchestration Engine** | AWS proprietary | Managed Kubernetes |
| **Setup complexity** | Simple | Complex |
| **Multi-cloud support** | No | Yes |
| **Launch types** | EC2, Fargate (serverless) | EC2, Fargate (serverless) |
| **Scaling** | AWS auto scaling | Kubernetes-native |
| **Pricing** | Free (pay only for resources) | $0.10/hour per cluster |
| **Security** | IAM roles, Security groups | Kubernetes RBAC + IAM roles for service (IRSA) |
| **Node Management** | Managed by AWS or EC2 | Self-managed worker nodes |
| **Best for** | Teams wanting simplicity & AWS integration. | Teams needing K8s flexibility & multi-cloud support. |

1. How does Docker integrate with AWS CI/CD pipelines?

* AWS EC2 can be used to host and deploy Docker Containers as part of a CI/CD pipeline using CodePipeline, CodeBuild, and EC2 instances.

**CI/CD Workflow Using EC2:**

1. **Code Commit:**

* Developers push code to AWS CodeCommit, GitHub, or Bitbucket.

1. **Build Docker Image:**

* **AWS CodeBuild** pulls the code, builds a Docker image, ad pushes it to Amazon ECR.

1. **Deploy to EC2:**

* A EC2 instance pulls the Docker image from ECR and runs the updated comtainer.
* Deployment can be automated with the AWS CodeDeploy.

**Steps to Deploy:**

* **Launch an EC2 instance:**
* Launch an EC2 instance with an Amazon Linux or Ubuntu AMI.
* **Install Docker:**
* SSH into the instance and install Docker:

**sudo yum update -y**

**sudo yum install docker -y**

**sudo service docker start**

* **Run the Docker Container:**
* Pull and run your docker image:

**Sudo docker run -d -p 80:80 your-docker-image**

* **Access the Application:**
* Use the public IP address of the EC2 instance to access the application.

1. What is the role of AWS Fargate in Docker container deployment?

* AWS Fargate is a serverless container service that allows you to run Docker Containers without managing EC2 instances.

**Key Roles:**

* **Serverless Container Management:** No need to manage EC2 instances.
* **Seamless Integration:** It works with the Amazon ECS and Amazon EKS for container orchestration.
* **Auto-Scaling:** It automatically scales containers based on workload.
* **Security & Isolation:** Each task runs in its own environment, improving security.
* **Cost-Efficient:** Pay only for the CPU and memory used by the running containers.

**How Fargate Works with Docker:**

* Create a Dockerfile and build the image.
* Push the image to Amazon ECR.
* Deploy the Fargate using ECS:
* Create an ECS task definition
* Run as Fargate Task.

1. How do you manage Docker container scaling on AWS?

* Scaling Docker containers on AWS can be achieved using auto-scaling policies, orchestration tools and serverless options.

**Scaling Docker containers on EC2 with Auto-scaling groups:**

* Run Docker container on EC2 instances and use Auto-scaling groups.
* **Steps:**
* Set up an EC2 Auto Scaling Group.
* Attach an Elastic Load Balancer (ALB) to distribute traffic.
* Configure scaling policies based on CPU/requests.

1. What is a Docker Compose file, and how can it be used with AWS?

* A docker compose file is YAML configuration file used to define and manage multi-container applications in Docker.
* It allows you to run multiple containers with a single command.
* It simplifies the orchestration, networking and scaling for multi-container applications.

**How it is used?**

* **Define services** in --docker-compose.yml
* **Run** docker-compose up –to start all containers.

**Common commands:**

* Start services: **docker-compose up -d**
* Stop services: **docker-compose down**
* Check logs: **docker-compose logs**

1. How do you monitor Docker containers in AWS?

* You can monitor docker containers in AWS using CloudWatch, CloudTrail, ECS/EKS built-in monitoring, and third-party tools.

**Monitoring Docker Containers on EC2:**

If running on EC2, use:

* **Docker stats command:**

Docker stats

* **CloudWatch Agent:**
* Install CloudWatch Agent:

sudo yum install amazon-cloudwatch-agent

* Configure /opt/aws/amazon-cloudwatch-agent.json:
* Start the agent:

sudo amazon-cloudwatch-agent-ctl -a fetch-config -m ec2 -c file:/opt/aws/amazon-cloudwatch-agent.json -s

1. How do you secure Docker containers on AWS?

* Securing Docker containers on AWS involves image security, network security, access controls and runtime protection.
* **Secure Docker Images:**
* Prefer official images from the DockerHub website.
* Avoid **:latest** tags and use immutable tags.
* Use image scanner to detect vulnerabilities.
* Use multi-stage build to remove build-time dependencies.
* **Secure Container Runtime:**
* Don’t run the container has the root user.
* Don’t install/configure things with the Dockerfile without understanding the potential risks.
* Use IAM roles & AWS secrets manager for secure access.

1. How do you optimize cost while running Docker containers on AWS?

* To reduce costs while running Docker containers on AWS, optimize compute resources, storage, networking and licencing.