AWS Elastic Container Registry (ECR) Notes

Overview:

- **Amazon ECR** (Elastic Container Registry) is a fully managed Docker container registry that makes it easy for developers to store, manage, and deploy Docker container images.
- It is integrated with AWS services like **Amazon ECS**, **Amazon EKS**, and **AWS Lambda**, enabling easy deployment of containerized applications.
- ECR supports private image repositories, providing security and access control.

Key Features:

1. **Private Repositories:**
    - ECR provides private Docker repositories where you can store your container images.
    - You can create multiple repositories and manage access via **IAM roles and policies**.
2. **Integration with CI/CD:**
    - Seamless integration with **AWS CodePipeline** and **AWS CodeBuild** to enable continuous integration and continuous delivery (CI/CD) pipelines.
3. **Security & Encryption:**
    - All images stored in ECR are encrypted at rest using **AWS KMS (Key Management Service)**.
    - IAM policies control who can access and modify the container images.
    - Supports **image scanning** to detect vulnerabilities in Docker images.
4. **Docker Integration:**
    - ECR is compatible with the Docker CLI, so developers can use familiar Docker commands (e.g., `docker push` and `docker pull`) to interact with repositories.
    - Provides Docker credential helpers for authenticating to private repositories.
5. **Highly Available:**
    - Amazon ECR is fully managed and highly available, ensuring images are accessible when needed.
6. **Version Control:**
    - ECR allows image versioning through **tags**. Each image pushed to ECR is assigned a unique **digest**.

How to Use AWS ECR:

1. **Create a Repository:**

Navigate to the ECR console or use the AWS CLI to create a repository.

```
aws ecr create-repository --repository-name my-repo
```

2. **Authenticate Docker to ECR:**

Use the AWS CLI to authenticate Docker to the ECR registry.

```
aws ecr get-login-password --region <region> | docker login --username
AWS --password-stdin <account-id>.dkr.ecr.<region>.amazonaws.com
```

3. **Push an Image to ECR:**

Tag the Docker image with your ECR repository URL.

```
docker tag my-image:latest
<account-id>.dkr.ecr.<region>.amazonaws.com/my-repo:latest
```

Push the image to ECR.

```
docker push <account-id>.dkr.ecr.<region>.amazonaws.com/my-repo:latest
```

4. **Pull an Image from ECR:**

Pull the image using Docker from the ECR repository.

```
docker pull <account-id>.dkr.ecr.<region>.amazonaws.com/my-repo:latest
```

Image Scanning for Vulnerabilities:

ECR provides the option to scan images for vulnerabilities by enabling **image scanning** at the repository level.

```
aws ecr start-image-scan --repository-name my-repo --image-id
imageTag=latest
```

Common Use Cases:

1. **Storing Custom Application Docker Images:**

   Ideal for teams using Docker to build and deploy custom applications in **ECS**, **EKS**, or directly on EC2.

2. **CI/CD Pipelines:**

   Integrates with AWS CodePipeline and CodeBuild for automating testing, building, and deploying containerized apps.

3. **Image Scanning for Security:**

   Use ECR's image scanning feature to ensure that container images are free from known vulnerabilities before deployment.

Pricing:

- **Storage Pricing:**

  Charged per GB of image storage.

- **Data Transfer Pricing:**

  Charges apply for data transferred out of ECR to the internet.

  Data transferred between ECR and services like ECS or EKS within the same region is free.