**LAB#8**

**Secure Container Images with ECR and Clair**

**\*\*Tasks\*\*:**

**- Push an image to ECR.**

**- Integrate Clair with ECR.**

**- Scan and analyse results.**

**- \*\*Documentation\*\*:**

**Emphasise the importance of scanning images for**

**vulnerabilities.**

**Container Images with ECR and Clair**

**Prerequisites:**

**Install kubectl**

**Install eks**

**Install awscli**

**Add access key**

**Add roles for cluster and node permissions**

**Add role ecr**

**dockerfile**

| **# Use the official BusyBox base image from Docker Hub FROM busybox:latest  docker-compose install  # Create a directory inside the container RUN mkdir -p /myapp  # Set this directory as the default working directory WORKDIR /myapp  # Create a sample file inside the container (for demonstration purposes) RUN echo "Hello, this is a sample file created in BusyBox." > sample.txt  # Specify the default command to run when the container starts** |
| --- |

| CMD ["sh"] |
| --- |

## **Push commands for push-image**

pull image

| aws ecr-public get-login-password --region us-east-1 | docker login --username AWS --password-stdin public.ecr.aws/i0r8m9k5 |
| --- |

| $ docker pull public.ecr.aws/i0r8m9k5/push-image:latest |
| --- |

To use Clair for vulnerability scanning, you'll need to have Clair installed locally to scan the Docker image you pulled from ECR. Here's a step-by-step guide on installing and using Clair:

Vi Docker-compose.yaml:

| Installing Clair Scanner Option 1: Using Docker Compose Create a docker-compose.yaml file:   version: '3' services:  clair:  image: quay.io/coreos/clair:v4.0.0  restart: always  container\_name: clair  ports:  - "6060:6060"  depends\_on:  - postgres   postgres:  image: postgres:12  restart: always  container\_name: postgres  environment:  POSTGRES\_PASSWORD: password  POSTGRES\_USER: postgres  volumes:  - clair\_db:/var/lib/postgresql/data  volumes:  clair\_db: |
| --- |

| clair-scanner -c http://localhost:6060 <your-local-image> |
| --- |

[**https://github.com/arminc/clair-scanner**](https://github.com/arminc/clair-scanner)

**Conclusion of the lab:**

**Scanning container images for vulnerabilities, whether through native solutions like ECR or utilizing tools like Clair, is crucial for bolstering the security posture of containerized environments.**

* **Early Vulnerability Detection: Image scanning enables early detection of vulnerabilities, reducing the risk of potential exploits and enhancing overall security within the container ecosystem.**
* **Compliance and Risk Mitigation: By adhering to security best practices and regulatory requirements, image scanning contributes to maintaining a compliant and secure environment, mitigating risks associated with known vulnerabilities.**
* **Integrated Security Measures: Leveraging tools like ECR and Clair for image scanning ensures a proactive approach to security, integrating vulnerability assessment seamlessly into the development and deployment lifecycle.**

**Incorporating image scanning as a standard practice in the containerization workflow significantly enhances the security posture of containerized applications. Continuous monitoring and remediation of vulnerabilities identified in image scans are essential for maintaining a resilient and secure container ecosystem.**