# DevOps Pipeline: From Git to Red Hat ACS

This document explains the DevOps flow process from Git to Jenkins Orchestrator, Red Hat Quay, and finally Red Hat ACS. It covers the entire CI/CD pipeline, security scanning, and troubleshooting common integration issues.

## Table of Contents

1. DevOps Flow Process  
2. Red Hat ACS CVE Scanning  
3. Application Visibility Issues  
4. Key Configuration Files  
5. Troubleshooting Steps

## DevOps Flow Process

The DevOps pipeline follows these key steps:

### Git Source Control

* - Developers commit and push code changes to Git repositories.
* - Repositories contain application code, infrastructure as code, and configurations.
* - Feature branches are created and merged into main/master through pull requests.

### Jenkins Orchestration

* - Jenkins monitors Git repositories for changes through webhooks.
* - Triggers the CI/CD pipeline, which includes:
* - - Code checkout from Git
* - - Building application artifacts
* - - Running automated tests
* - - Static code analysis
* - - Security scans
* - - Container image building

### Red Hat Quay Container Registry

* - Jenkins builds and pushes container images to Red Hat Quay.
* - Quay performs automated security scanning and tagging.
* - Provides vulnerability reporting and policy enforcement.

### Red Hat Advanced Cluster Security (ACS)

* - ACS integrates with Quay to provide deeper security insights.
* - Performs continuous scanning of running containers.
* - Enforces security policies and monitors runtime behavior.

## Red Hat ACS CVE Scanning

Red Hat ACS performs vulnerability scanning to detect security risks in container images.

### Scanning Process

* - ACS automatically scans container images both at rest and in runtime.
* - Uses vulnerability databases like Red Hat Security Data, NVD, and language-specific feeds.
* - Detects Common Vulnerabilities and Exposures (CVEs) in OS packages and dependencies.

## Application Visibility Issues in ACS

Several misconfigurations can prevent an application from appearing in ACS.

### Configuration Misalignment

Incorrect labels or metadata prevent ACS from tracking the application.

### Pipeline Communication Gaps

Failures in webhooks or network policies prevent ACS from detecting deployments.

### Scanning Configuration Issues

Improper repository inclusion/exclusion settings affect visibility.

### Permission and Access Control

ACS needs permissions to read container image metadata and deployment configurations.

### Deployment Environment Mismatch

The application is deployed in an environment that ACS is not monitoring.

## Key Configuration Files & Common Issues

### Jenkins Pipeline (Jenkinsfile)

Defines the CI/CD pipeline, including build, test, and image push steps.

### Kubernetes Deployment Manifest (deployment.yaml)

Specifies application deployment details, including labels and image references.

### Red Hat Quay Configuration (config.yaml)

Manages Quay registry settings, security scanning, and repository visibility.

### ACS Configuration Files (centralconfig.yaml, scannerpolicy.yaml)

Handles security scanning policies and integration settings.

## Troubleshooting Steps

* - Verify Jenkins build logs to ensure images are correctly pushed to Quay.
* - Confirm images appear in Quay with proper tags.
* - Check Quay’s security scanner logs for successful scans.
* - Ensure ACS has valid authentication credentials for Quay.
* - Review ACS scanner logs for failed scan attempts.

## Conclusion

Ensuring a smooth DevOps flow requires proper integration between Git, Jenkins, Quay, and ACS. By maintaining consistency in configuration files, aligning security policies, and troubleshooting errors proactively, organizations can achieve a seamless CI/CD process with robust security enforcement.