Doco bits

# Logging

OpenShift components produce different log types. Container logs are generated from both OpenShift component containers and application workload containers and can be used for application monitoring and troubleshooting. The OpenShift and Kubernetes API servers and the Red Hat CoreOS operating system produce audit logs for security auditing and incident detection purposes.

The audit logs produce a lot of data, much of relating to internal processes within the cluster and much of the audit data relates to read activity. Red Hat recommends using the Cluster Logging operator filter options to filter the audit data so that only audit data related to an update or delete activity be forwarded to the logging endpoint. Further filtering can be implemented to exclude logs from actions instigated by internal service accounts.

ACIC utilises Splunk for audit log collection. The following table shows the logs being forwarded to Splunk from each cluster.

| **Cluster** | **Logging Source** | **Destination** | **Notes** |
| --- | --- | --- | --- |
| SDC Hub | Application logs | none |  |
| SDC Hub | Infrastructure logs | none |  |
| SDC Hub | Audit logs | Splunk | Filtered to exclude read (get/list/watch) API calls |
| PDC Hub | Application logs | none |  |
| PDC Hub | Infrastructure logs | none |  |
| PDC Hub | Audit logs | none | Filtered to exclude read (get/list/watch) API calls |
| Development | Application logs | Loki | Filtered to only capture OpenShift Pipelines task pod logs |
| Development | Infrastructure logs | none |  |
| Development | Audit logs | none |  |
| Test ES | Application logs |  |  |
| Test ES | Infrastructure logs |  |  |
| Test ES | Audit logs |  |  |
| Test CS | Application logs |  |  |
| Test CS | Infrastructure logs |  |  |
| Test CS | Audit logs |  |  |
| SDC Prod | Application logs |  |  |
| SDC Prod | Infrastructure logs |  |  |
| TDC Prod | Audit logs |  |  |
| PDC Prod | Application logs |  |  |
| PDC Prod | Infrastructure logs |  |  |
| PDC Prod | Audit logs |  |  |

*Table X: Logging targets*

The OpenShift Pipelines logs from the development cluster are also to be retained. These logs are forwarded to an S3 bucket accessible from the development enclave via the Red Hat Loki operator. The OpenShift Logging configuration captures the application logs from the tasks pods and forwards these to the Loki endpoint.

Retention TBD

# ACM Governance Policies

ACM policies are a method of scanning and enforcing standards or configuration on OpenShift clusters. Policies use its own custom resource definition to describe desired state and whether to inform or enforce the desired state via a policy remediation action for a policy.

ACM policies require a destination namespace to store any defined policies on the hub cluster. A namespace called rhacm-policies was created on the hub cluster using a CRD:

The ACM deployment currently uses ACM policies for two purposes:

Using a policy to ensure that the Compliance operator is installed on all OpenShift clusters.

Using a policy to ensure that the Compliance operator is configured correctly to scan an OpenShift cluster against the Essential 8 policy.

An important feature of Red Hat ACM is its advanced policy and governance capabilities. Red Hat ACM allows administrators to define policies that govern the deployment and configuration of Kubernetes resources, ensuring that applications are deployed in a secure and compliant manner. Additionally, Red Hat ACM provides real-time compliance and risk assessments, giving the ability to quickly identify and address any security or compliance issues.

Red Hat Advanced Cluster Management (ACM) offers robust policy and governance capabilities. It enables administrators to define policies that govern the deployment and configuration of Kubernetes resources, ensuring applications are deployed securely and compliantly. ACM also provides real-time compliance and risk assessments, allowing for quick identification and resolution of security or compliance issues.

ACM policies utilize a custom resource definition to describe the desired state and specify the remediation action (inform or enforce) for that state. A dedicated namespace, rhacm-policies, is created on the hub cluster using a Custom Resource Definition (CRD) to store these policies.

Currently, ACM policies are used for two primary purposes:

* Applying the NTP configuration to all clusters.
* Something else TBD

# Container scanning

## Active images

Describe container scanning of active images using red hat acsContainer scanning of active images using Red Hat Advanced Cluster Security (ACS) for Kubernetes is a critical aspect of maintaining a secure containerized environment. Red Hat ACS provides robust capabilities for identifying vulnerabilities and misconfigurations within container images, both at rest and in runtime.

**Key aspects of container scanning with Red Hat ACS:**

* **Image Registry Integration:** ACS integrates with various image registries (e.g., Quay, Docker Hub, private registries) to scan images as they are pushed or pulled. This "shift-left" approach ensures that vulnerabilities are identified early in the development lifecycle, before deployment to production.
* **Vulnerability Management:** ACS leverages a comprehensive vulnerability database to identify known CVEs (Common Vulnerabilities and Exposures) within container images. It provides detailed information about each vulnerability, including severity, affected components, and recommended remediation steps.
* **Policy Enforcement:** Organizations can define custom security policies in ACS to dictate acceptable security postures for container images. These policies can enforce controls such as:
  + **Denying deployments of images with critical vulnerabilities:** Preventing unsecure images from running in the cluster.
  + **Requiring specific base images:** Ensuring that only approved and hardened base images are used.
  + **Scanning for sensitive data:** Identifying potential exposure of sensitive information within images.
  + **Checking for misconfigurations:** Detecting insecure configurations that could be exploited.
* **Runtime Security:** Beyond static image scanning, ACS also monitors active containers for anomalous behavior and potential threats. If an image that was initially deemed secure at rest exhibits suspicious activity at runtime (e.g., unexpected network connections, unusual process execution), ACS can detect and alert on these deviations.
* **Compliance and Reporting:** ACS assists in meeting regulatory compliance requirements by providing detailed reports on image security posture, vulnerability trends, and policy violations. This helps organizations demonstrate due diligence in securing their container environments.
* **Integration with CI/CD Pipelines:** Embedding ACS scanning into continuous integration and continuous delivery (CI/CD) pipelines automates the security checks, making security an integral part of the software development process. This ensures that every image built and deployed undergoes thorough security scrutiny.
* **Risk Prioritization:** ACS helps security teams prioritize vulnerabilities by considering factors such as exploitability, impact, and whether the vulnerability exists in active, running containers. This allows teams to focus on the most critical risks first.

By leveraging Red Hat ACS for container scanning, organizations can significantly reduce their attack surface, mitigate security risks, and ensure the integrity and compliance of their OpenShift Container Platform deployments.

Image scanning in build pipelinesImage Scanning in Build Pipelines

Integrating image scanning directly into build pipelines is a crucial security practice for modern software development. This process involves analyzing container images for known vulnerabilities, misconfigurations, and other security risks **before** they are deployed to production environments. By embedding scanning at this early stage, organizations can identify and remediate issues proactively, significantly reducing their attack surface and preventing the introduction of compromised images into their systems.

**Key Benefits:**

* **Early Detection and Remediation:** Catching vulnerabilities during the build phase is far more efficient and cost-effective than discovering them after deployment. It allows developers to address issues immediately, preventing them from propagating further down the development lifecycle.
* **Automated Security Enforcement:** Integrating scanning into CI/CD pipelines automates security checks, ensuring that every image built undergoes scrutiny. This eliminates manual oversight and enforces security policies consistently.
* **Reduced Risk of Supply Chain Attacks:** By scanning all components and dependencies within an image, organizations can mitigate the risk of supply chain attacks, where malicious code is injected into upstream software or libraries.
* **Compliance and Governance:** Many regulatory frameworks and industry standards mandate vulnerability scanning. Automated image scanning helps organizations meet these compliance requirements and maintain a strong security posture.
* **Improved Developer Productivity:** By providing immediate feedback on security issues, developers can quickly rectify problems without waiting for security team reviews or post-deployment scans. This streamlines development workflows and enhances overall productivity.

**How it Works:**

During the build process, after a container image is created, the scanning tool is invoked. This tool typically performs the following actions:

1. **Vulnerability Database Lookup:** The scanner compares the image's components (operating system packages, libraries, application dependencies) against continuously updated databases of known vulnerabilities (CVEs – Common Vulnerabilities and Exposures).
2. **Configuration Analysis:** It checks for common security misconfigurations, such as exposed ports, default credentials, or unnecessary privileges.
3. **Malware Detection:** Some advanced scanners can also identify embedded malware or suspicious files within the image.
4. **License Compliance:** Scanners can identify the licenses associated with open-source components, helping to ensure compliance with legal requirements.

**Integration with OpenShift Container Platform:**

OpenShift Container Platform, with its robust CI/CD capabilities (e.g., using OpenShift Pipelines based on Tekton), provides an ideal environment for integrating image scanning. Scanners can be incorporated as a step within a Tekton Task or Pipeline, triggered automatically upon successful image build. The results of the scan can then be used to:

* **Fail the Build:** If critical vulnerabilities are detected, the build process can be automatically halted, preventing the insecure image from being pushed to the registry.
* **Generate Reports:** Comprehensive reports can be generated, detailing identified vulnerabilities, their severity, and recommended remediation steps.
* **Notify Stakeholders:** Security teams and developers can be automatically notified of scan results, facilitating rapid response and collaboration.

By fully embracing image scanning in build pipelines, organizations can significantly strengthen their security posture, accelerate their development cycles, and ensure that only secure and compliant container images are deployed into their OpenShift environments.

## Inactive images

Explain what is an active image

### ACS

Describe container scanning of inactive images using red hat acsWhen implementing a robust security posture for your OpenShift Container Platform, it is crucial to address the risk posed by inactive container images. These images, while not currently deployed or running, can harbor vulnerabilities that could be exploited if they were to be instantiated in the future. Red Hat Advanced Cluster Security (ACS) for Kubernetes provides comprehensive capabilities to address this concern through its container scanning features.

**Scanning Inactive Images with Red Hat ACS:**

Red Hat ACS extends its image scanning capabilities beyond actively deployed images to include those residing in your registries but not currently in use. This proactive approach ensures that your entire image inventory is regularly assessed for security risks. Here's a breakdown of how ACS facilitates this:

* **Registry Integration:** ACS seamlessly integrates with various container registries, including Quay.io, Docker Hub, and private registries, allowing it to access and scan images regardless of their location.
* **Automated Scanning:** You can configure ACS to automatically scan new images as they are pushed to your registries and to periodically re-scan existing inactive images. This ensures continuous vulnerability assessment without manual intervention.
* **Vulnerability Detection:** ACS utilizes a comprehensive vulnerability database to identify known Common Vulnerabilities and Exposures (CVEs) within your inactive images. It analyzes image layers, packages, and dependencies to provide a detailed report of identified risks.
* **Policy Enforcement:** With ACS, you can define policies that dictate acceptable security standards for your images. These policies can be applied to inactive images, flagging those that do not meet your organization's security requirements. For example, you could establish a policy that blocks the deployment of any image with high-severity vulnerabilities older than a certain number of days, even if it's currently inactive.
* **Risk Prioritization:** ACS helps you prioritize vulnerabilities by assigning severity levels and providing context on potential exploitability. This allows your security teams to focus on the most critical risks first, even within your inactive image inventory.
* **Reporting and Alerting:** ACS generates detailed reports on scanned images, highlighting identified vulnerabilities and policy violations. It can also be configured to send alerts to relevant teams when critical issues are detected in inactive images, prompting timely remediation.
* **Compliance Adherence:** By consistently scanning inactive images, ACS assists organizations in maintaining compliance with various industry regulations and internal security policies that mandate vulnerability management across the entire software supply chain.

**Benefits of Scanning Inactive Images:**

* **Reduced Attack Surface:** By identifying and remediating vulnerabilities in inactive images, you significantly reduce the potential attack surface of your OpenShift environment.
* **Proactive Security:** This approach shifts security from a reactive to a proactive stance, allowing you to address issues before they become exploitable in production.
* **Improved Security Posture:** Regular scanning of all images, active or inactive, contributes to a stronger overall security posture for your containerized applications.
* **Enhanced Compliance:** Meeting regulatory requirements often involves demonstrating comprehensive vulnerability management, which includes inactive images.
* **Efficient Resource Allocation:** By prioritizing risks, security teams can allocate their resources more effectively, focusing on the most impactful vulnerabilities first.

In summary, leveraging Red Hat ACS for container scanning of inactive images is an essential practice for maintaining a secure and compliant OpenShift Container Platform. It provides the visibility and control needed to mitigate risks associated with dormant software components, ultimately contributing to a more resilient and secure application environment.

### Clair

Describe container scanning of images in Quay using clairOpenShift Container Platform leverages Quay's integrated security features, specifically Clair, for robust container image scanning. This process is crucial for identifying vulnerabilities within container images before and during their deployment in the OpenShift environment.

**How Container Scanning with Clair in Quay Works:**

1. **Image Upload/Push to Quay:** When a container image is pushed to a Quay repository, it triggers the scanning process. Quay acts as a central registry for storing and managing these images.
2. **Clair Integration:** Clair is a highly scalable open-source vulnerability scanner that integrates seamlessly with Quay. Upon image push, Clair analyzes the image layers.
3. **Vulnerability Database:** Clair maintains an up-to-date database of known vulnerabilities (CVEs - Common Vulnerabilities and Exposures) from various sources, including public vulnerability databases and vendor advisories.
4. **Layer-by-Layer Analysis:** Clair breaks down the container image into its individual layers. It then compares the software packages and their versions within each layer against its vulnerability database.
5. **Vulnerability Reporting:** After the scan, Clair generates a detailed report outlining any detected vulnerabilities, including:
   * **CVE ID:** A unique identifier for the vulnerability.
   * **Severity:** The impact level of the vulnerability (e.g., critical, high, medium, low).
   * **Affected Package:** The specific software package with the vulnerability.
   * **Recommended Fix:** Information on how to mitigate the vulnerability, often including recommended package updates.
6. **Quay UI and API Integration:** The scan results are presented directly within the Quay user interface, providing a clear overview of the image's security posture. These results are also accessible via Quay's API, enabling integration with other security tools and CI/CD pipelines.

**Benefits of Using Clair for Container Scanning in OpenShift:**

* **Early Vulnerability Detection:** Identifying vulnerabilities early in the development lifecycle (Shift Left security) reduces the risk and cost of remediation.
* **Automated Security Checks:** Automating the scanning process ensures that all images are consistently checked for known vulnerabilities without manual intervention.
* **Improved Security Posture:** By regularly scanning images and acting on the reported vulnerabilities, organizations can significantly enhance the overall security posture of their OpenShift deployments.
* **Compliance:** Facilitates compliance with various security regulations and industry standards that require vulnerability management for software components.
* **Integration with OpenShift Workflows:** While Quay manages the image scanning, the results can inform decisions within OpenShift, such as preventing the deployment of images with critical vulnerabilities through admission controllers or policy engines.
* **Supply Chain Security:** Provides visibility into the security of the software supply chain by scanning base images and subsequent layers for potential weaknesses.

**Best Practices for Container Scanning:**

* **Scan Frequently:** Implement regular, automated scanning as part of the CI/CD pipeline.
* **Prioritize Fixes:** Address high and critical severity vulnerabilities first.
* **Use Trusted Base Images:** Start with secure, minimal base images from trusted sources.
* **Monitor for New Vulnerabilities:** Continuously monitor Quay and Clair for newly discovered vulnerabilities that might affect existing deployed images.
* **Integrate with Policy Enforcement:** Utilize OpenShift's admission controllers to enforce policies based on scan results, preventing the deployment of non-compliant images.
* **Regularly Update Clair's Database:** Ensure that Clair's vulnerability database is regularly updated to catch the latest threats.