KSPM Documentation

1 | Kubernetes Security Posture Management (KSPM)

The Kubernetes Security Posture Management (KSPM) capability, driven by the Kubernetes Connector component, is positioned as a core security module within Cortex Cloud. KSPM is a lightweight solution for Kubernetes posture management that automatically discovers assets, enforces policies, and scans for vulnerabilities, malware, secrets, and misconfigurations across the environment.

The Cortex Cloud KSPM offering focuses on deep security posture and compliance checks:

- Inventory and visibility: KSPM provides full visibility into your Kubernetes cluster and resources, including namespaces, nodes, and workloads. The KSPM dashboard provides a visual overview including inventory insights and cluster protection coverage to show which cluster have no protection solution deployed.
- Compliance and misconfiguration detection: KSPM leverages hundreds of out-of-the-box KSPM rules. It detects compliance violations and misconfigurations using built-in and custom rules mapped to compliance controls. Compliance checks include CIS Benchmarks for both managed and unmanaged Kubernetes distributions. Custom rules are supported using Rego for the Kubernetes Connector, and Python for XDR Agent endpoints.
- Vulnerability, malware, and secret scanning: KSPM receives critical security information related to vulnerabilities, malware, secrets, and other available scanners. The KSPM dashboard includes metrics for malware detected and secrets detected in clusters
- Policy enforcement and control: By enabling an admission controller, you can ensure that all resources created within the cluster adhere to the desired security and governance standards, enhancing the overall security posture of your environment. The admission controller intercepts requests to the Kubernetes API server before they are persisted, allowing enforcement of access control, image assurance, and security configurations.
- Centralized policy management: The offering provides centralized Cloud Workload Policy management and enforcement at runtime when the admission controller is utilized.

2 | KSPM dashboard

IMPORTANT:

Users can access all information on the dashboard when their user access is scoped to view All assets or assigned to the Instance Administrator role. Otherwise, users with granular scoping set to No assets or Select asset groups will have limited access to the dashboard. For more information on Scope-Based Access Control (SBAC), see Manage user scope in the Cortex Cloud Runtime Security Documentation or Manage user scope in the Cortex Cloud Posture Management Documentation.

The KSPM dashboard provides a centralized overview of your entire Kubernetes environment, enabling you to quickly identify risks and prioritize remediation actions. The KSPM dashboard is included in the predefined dashboards.

To access the KSPM dashboard, select Dashboards & Reports → Dashboard. From the dashboard header, a drop-down menu lists all available predefined and custom dashboards. Select KSPM. You can use the filtering options at the top of the dashboard to focus on specific cloud accounts and clusters.

The dashboard consolidates critical security and operational data into the following widgets, allowing you to assess your security posture at a glance:

- Kubernetes Assets Overview: Obtain a high-level inventory of your Kubernetes environment, displaying the total count of clusters, namespaces, nodes, and workloads. Click on the drop-down icon for each asset to see a detailed breakdown of the Kubernetes platform type or workload. Click on an asset type to go to the Kubernetes Resources page, filtered according to that asset type.

- Cluster Protection Coverage: This widget shows the percentage of clusters that are fully protected, protected with posture management, protected with realtime management, or unprotected. Identify your protection coverage gaps so that you can prioritize riskiest clusters and onboarding of unprotected clusters. Click on the more options icon to see all clusters in the Kubernetes Connectivity Management page.

- Malware Detected: A list of cluster names and the total count of malware detected within them to allow for quick identification and response to active threats. Click on the more options icon to go to the Policy Management page.

- Cases and Issues: Displays the total number of cases and issues currently opened across your Kubernetes environment. Click on the more options icon to go to the Cases or Issues pages.

- Top Clusters by Vulnerabilities: Kubernetes clusters are ranked based on the number of high-severity vulnerabilities, ranked by CVSS score. Click on the more options icon to see all clusters in the Kubernetes Connectivity Management page or to go to the Vulnerability Policies Management page.

- Secretes Detected in Clusters: Shows the the number of unsecured secrets found within your clusters. Click on the more options icon to go to the Policy Management page.

3 | Kubernetes Clusters

The Kubernetes Clusters page provides a macro view of the entire cluster environment discovered by Cortex Cloud. The inventory view displays details about each cluster, including the cluster name, Kubernetes platform, Kubernetes version, cloud account, and whether the Kubernetes

Connector is deployed. You can access the Kubernetes Clusters page by navigating to Inventory → All Assets → Compute → Kubernetes Clusters.

> **NOTE:**
>
> Data for the Resource Explorer and Vulnerabilities tabs are populated with data once the posture management solution is deployed.

Selecting any cluster in the inventory opens the detailed asset card with the following tabs:

- Overview: Summarizes the high-level identifying details and security posture of the cluster. Cluster details including asset ID, provider, cloud region, and asset groups. If the Kubernetes Connector is deployed on this cluster, the connector details are also displayed. A clickable relationship graph allows you to visualize and understand the full context and dependencies of the cluster.
- Resource Explorer: Provides granular visibility into the specific assets deployed within the cluster. It includes a detailed list of discovered resources, including the namespace (when applicable), asset name, asset type, and category type.
- Identity: Displays the list of identities that can access this Kubernetes cluster.
- Configurations:Presents a list of security configuration issues detected within the Kubernetes cluster. For each issue, you can view the severity, name, category, and creation date. You can also view the asset configuration JSON.
- Vulnerabilities: Details all discovered vulnerabilities found within the container images running in the cluster, as well as for the host operating systems of your Kubernetes nodes.

4 | Kubernetes Resources Inventory

The Kubernetes Resources page provides a detailed, micro view of the individual components deployed inside your connected Kubernetes clusters. The Kubernetes resources are tracked and managed within the broader Unified Asset Inventory (UAI). You can access the Kubernetes Resources page by navigating to Inventory → All Assets → Compute → Kubernetes Resources.

The Kubernetes Resource page is a very granular inventory of Kubernetes objects discovered by the Kubernetes Connector:

- Workloads: Deployments, ReplicaSets, StatefulSets, DaemonSets, Jobs, and CronJobs
- Identity: ServiceAccounts, ClusterRoles, Roles, RoleBindings, and ClusterRoleBindings
- Network/Configuration: Services, Endpoints, Ingresses, NetworkPolicies, ConfigMaps, and Secrets

Use the filter at the top of the page to customize precisely which resources you want to view. Selecting any resource in the inventory opens the detailed asset card. The asset card provides a detailed overview, including asset properties, such as its ID, provider, region, and the cluster it belongs to. It also includes a clickable relationship graph to visualize and understand the full context and dependencies of the resource within the cluster. The asset card also includes the resource's code and vulnerability findings.

5 | Onboard the Kubernetes Connector

Abstract

To onboard your Kubernetes cluster, choose the capabilities that fit your needs and download the Helm chart values. Install the Helm charts in your Kubernetes environment to grant Cortex Cloud permissions to collect the data.

Follow this wizard to deploy your Kubernetes Connector. The Kubernetes onboarding wizard is designed to facilitate the seamless setup of Kubernetes data into Cortex Cloud. The guided experience requires minimal user input; simply select the capabilities that fit your needs and download the custom installer file. For full control of the setup, you can use the advanced settings. Based on the onboarding settings, Cortex Cloud then creates a custom installer file for running in your Kubernetes environment. This file, once executed in your Kubernetes environment, grants Cortex Cloud the necessary permissions to collect the data. The installer file must be executed in your Kubernetes environment to complete the onboarding process. The connector then appears in Kubernetes Connectors.

1. Select Settings → Data Sources.

2. Select Add Data Source.

3. On the Add Data Sources page, search for and select Kubernetes and click Connect.

4. In the Kubernetes Connect onboarding wizard, enable the solutions that fits your needs:

    - Posture Management: (Enabled by default) A lightweight posture management solution for continuous discovery, policy enforcement, and proactive scanning of vulnerabilities, secrets, malware, compliance, and misconfigurations.

    - Realtime Protection: A solution that monitors workloads in real time to detect and block malicious activity, instantly preventing attacks as they happen.

5. (Optional) Click Edit to configure advanced settings and then click Apply Changes:

- Posture Management:

Setting	Notes
Scan Cadence (Hours)	Define how often to scan (from every one to 24 hours). Default is 12 hours.
Policy Enforcement by the Admission Controller	Select to allow enforcement policies to be configured, ensuring that only compliant resources are admitted into the cluster.

- Realtime Protection:

Setting	Notes
Node Selector	Enter node labels to have the agent run on nodes that match the node labels.
Run on all nodes (Including Master)/Run only on master node	
Deployment Platform	Select the Kubernetes deployment platform: ◦ Standard ◦ Bottlerocket OS ◦ Google GCOS ◦ OpenShift

6. (Optional) Click Edit Profile to customize the Kubernetes Connector's profile:

Setting	Notes
Profile Name	A profile name is automatically generated including the date and time of creation. You can manually change the profile name.
Version	Select which version of the Kubernetes Connector to install.
Cluster Resource Identifier	(Optional) Enter the Kubernetes cluster resource identifier. If you do not specify the resource identifier, the installer will identify the cluster on its own.
Namespace	Enter the name for the Kubernetes namespace. The default is "panw".
Proxy Gateway	Enable this option if network traffic between Cortex Cloud and your Kubernetes cluster must route through a proxy gateway. Enter the following details: • Proxy IP: The full IP address and port number for your HTTP proxy server. For example: `192.168.1.1:8080` • Authentication: Select None or Basic. Enter the username and password for a proxy user account that has permission to pass traffic to the Kubernetes cluster. **NOTE:** Basic authentication is only supported in Posture Management. If deploying Realtime Protection, select None .

Setting	Notes
Auto Upgrade	Enable Auto Upgrade to ensure the Kubernetes Connector and its installed capabilities are automatically updated to a newer version when available. This minimizes manual maintenance and ensures continuous access to the latest features and security patches. Select the Upgrade Strategy: • Latest Available Version (GA): Automatically upgrade to the newest version as soon as it is released to gain immediate access to all new features. • One release before the latest one (N-1): Maintain a policy to always remain one version behind the latest available release. Select Advanced to customize the upgrade schedule. Define whether to be upgraded immediately or to delay the upgrade by a specified number of days. You can then specify the preferred day and time for the upgrade to be applied.

7. Click Generate.
8. To complete the onboarding of the Kubernetes Connector, you must download the Helm chart values `values.yaml` and run it in your Kubernetes environment: `helm repo add cortex https://paloaltonetworks.github.io/cortex-cloud --force-update`
9. Install the Helm charts in your Kubernetes environment: `helm upgrade --install konnector cortex/konnector --wait-for-jobs --create-namespace --namespace panw --values <profile-name>.values.yaml`
10. Verify the deployment succeeded when you see "Status: Deployed"

    When the Kubernetes Connector is deployed, the initial discovery scan is started and the connector appears in Data Sources → Kubernetes → Kubernetes Connectors.

6 | Kubernetes connectivity management

The Kubernetes Connectivity Management provides visibility into all the Kubernetes clusters in your cloud environment along with its platform type, cloud account, Kubernetes version, and whether the Kubernetes Connector is installed. Clicking on a cluster opens the asset card for that cluster. If a Kubernetes cluster has the Kubernetes Connector deployed on it, you have increased visibility into all entities running on the cluster.

To access the Kubernetes cluster details, select a Kubernetes cluster. In the asset card, click the menu button and select Open Cluster Details.

The Overview tab presents a high-level summary of the Kubernetes cluster. This includes cluster properties, any associated cases, issues, or findings, and details regarding the Kubernetes Connector.

The Resource Explorer tab displays every single asset on any Kubernetes cluster with the Kubernetes Connector installed. You can filter according to Kubernetes namespace and asset category. This comprehensive visibility enhances your Kubernetes security by enabling effective vulnerability tracking, since you know precisely which containers and images are running. It also provides a clear audit trail of all active components within your environment. The increased visibility into your Kubernetes assets also makes it easier to manage and track your assets. This can help you to optimize your Kubernetes deployments and reduce cloud costs.

7 | Manage Kubernetes Connector instances

Abstract

You can manage the Kubernetes Connector instances on the Data Sources page. You can check the status, edit or delete Kubernetes Connector instances.

1. Select Settings → Data Sources.
2. Find the Kubernetes instance by clicking on the Kubernetes name or using the Search field.
3. In the row for the Kubernetes instance, click View Details. The Kubernetes Connectors page is displayed with all deployed Kubernetes Connectors. To view all Kubernetes clusters, including ones that are not yet deployed, go to the Kubernetes Connectivity Management page.
4. In the Kubernetes Connectors page, click on a cluster name to open the details pane for that instance.
5. You can perform the following actions on each Kubernetes Connector instance:

Action	Instructions
Open Cluster Details	In the details pane, click the more options icon and select Open Cluster Details. The Asset Card for that Kubernetes cluster is displayed.
Edit Connector	In the row for the Kubernetes instance, right-click and select Edit. Alternatively, in the details pane, click the more options icon and select Edit Connector. In Edit Kubernetes Connector, edit the configurations and click Apply Changes.You must execute the updated template in the Kubernetes environment for the configuration changes to be applied.
Delete Connector	In the row for the Kubernetes instance, right-click and select Delete. Alternatively, in the details pane, click the more options icon and select Delete Connector. To remove the connector, you must manually run Kubernetes commands to delete the resources in the Kubernetes environment. The commands are listed here.

Kubernetes Connectivity Management

Navigate to Settings → Data Sources and find the Kubernetes instances by clicking on the Kubernetes name or using the Search field. In the Kubernetes Connectors page, click Kubernetes Connectivity Management to view all detected Kubernetes clusters. Here, you can check if a cluster is connected, view the status, and see the connector version. When a new version of the Kubernetes Connector is available, you can update it here.

> **NOTE:**
>
> After uninstalling the Kubernetes connector, the connector status updates to Not connected 48 hours after the uninstall process is initiated.

8 | Cloud Workload Policies and Rules

Cloud Workload Policies and Rules help organizations maintain security compliance, prevent misconfigurations, and reduce risks across cloud environments.

- Cloud Workload Policies define organizational security objectives by combining detection logic with preventive actions across selected asset scopes. Policies can generate issues and proactively block misconfigurations before they reach runtime, ensuring workloads remain compliant with security requirements throughout the Software Development Life Cycle (SDLC). They leverage identified security risks and enforce controls at the right stages of development and operations, such as during CI pipelines or in runtime environments.

- Cloud Workload Rules define the detection logic for misconfigurations and their applicable asset types, specifying the criteria and conditions used to identify security risks. These rules can be selected and enforced through Misconfiguration Policies within the designated policy asset scope.

Together, Policies define which risks must be addressed and what actions to take, while Rules specify how those risks are detected through precise logic and conditions.

PREREQUISITE:

Users need View/Edit RBAC permissions (under Policies → Compute Policies) or the Instance Administrator role to view, edit, and modify Cloud Workload Policies policies.

IMPORTANT:

Users with SBAC granular scoping (in addition to the RBAC permissions required for Cloud Workload Policies) can only view Cloud Workload Policies, when their access is scoped to any of the available options: All assets, No assets, or Select asset groups. For more information on granular scoping, see Manage user scope. When no SBAC restriction is applied, the user's access is determined solely by their RBAC permissions.

8.1 | How policies and rules work together

Cloud Workload Policies serve as enforcement mechanisms that govern the responses to the identified findings, whereas *Cloud Workload Rules* establish the criteria for evaluation but do not initiate any actions unless incorporated within a policy.

In the absence of an associated policy, rules exist solely as evaluative criteria and do not generate alerts or trigger any response actions. Policies determine how findings from rules are escalated to issues or preventive measures.

8.2 | Cloud Workload Policies

Cloud Workload Policies help you prevent and manage security violations in your cloud runtime instances. They enable you to apply detection logic to specific asset groups at the desired SDLC stage, and define what action needs to be taken if the conditions are met.

Depending on the nature of the security violation, a Cloud Workload Policy allows you to

- *Prevent the violation*. Enable proactive prevention of the violation. For example: Block an S3 bucket deployment that is open to the public.
- *Create an issue*. Create an issue when violation is seen. For example: Create an issue when an AWS credential file is found on a Linux server.

> **NOTE:**
>
> Issues are automatically resolved when the finding is no longer applicable to the asset or when the affected asset is removed from the inventory.
>
> For more details on Prevent and create issues, see Cloud Workload Preventive Action.

A Cortex Cloud Workload Policy has the following elements:

- SDLC Evaluation Stage: The SDLC stage at which the policy is applied and evaluated. Depending on the policy type, one or more of the following stages may be available:
  - CI: The stage during which a pipeline builds the artifact. After building the artifact, the pipeline pushes it to a registry.
  - Deploy: The stage when the artifact is pushed to a cloud instance for running.
  - Runtime: The stage when the artifact is running on a cloud instance.
- Rule (Conditions): The logical conditions that will trigger the evaluation of this policy.
- Scope: A filter specifying which assets the rule applies to.
- Action: The response triggered when the rule evaluates successfully (only when part of a policy). Based on the rules included in the policy, it can create an issue or prevent the security violation.

8.2.1 | Types of Cloud Workload Policies

- Misconfiguration policies: Enables you to assess various workloads for misconfigurations against relevant security standards and your organization's security guidelines. You can include both predefined and custom rules in these policies to either prevent violations or create issues for violations.
- Malware policies: Enable you to detect and manage malicious files within cloud workloads. These policies analyze files based on predefined parameters such as file name, path, size, and detection method.
- Secret policies: Enable you to identify and protect sensitive information—such as API keys and credentials—within workloads.
- Trusted Image policies: Enable you to ensure the authenticity, integrity, and security of container images and VMs deployed into your Kubernetes environments. This includes actions such as limiting allowed image sources, mitigating possible image tampering, and more.

8.2.1.1 | Trusted image cloud workload policies

Abstract

Trusted image policies ensure the integrity and security of container images and VMs deployed into Kubernetes environments. Using these policies, you can be assured that your images are from a trusted source, are built on approved and validated base images, and are free from unauthorized modifications.

Trusted image policies ensure the integrity and security of container images and VMs deployed into Kubernetes environments. This topic details the policy enforcement logic that Cortex uses to determine if an image is trusted, what action to take, and which issues to generate.

Images are evaluated to see if they:

- Match the trusted image policy criteria.
- Should be allowed or prevented, based on policy criteria and scope.
- Should trigger the creation of security or posture issues when untrusted.

> **NOTE:**
>
> Registry scanning is for finding problems that are objectively issues regardless of context, organization, or scope. Trust, however, is subjective. Depending on the scope and other factors, an issue may or may not be problematic, and can change over time depending on the context.

Trusted image policy actions

When defining a trusted image policy, you determine what actions should be taken if an image is not trusted.

- Create an Issue. The image is allowed to run, but a violation is recorded as a posture and/or security issue. For the purposes of this documentation, we refer to this as an **issue-only** action.

  The primary purpose of trusted image policies with the issue-only action is auditing and compliance. These violations are identified by Cortex periodic scans.

  The issue-only action assesses trust across a range of assets, including Kubernetes workloads (which are cloud-agnostic, supporting AWS, Azure, GCP, OCI), virtual machines (VMs).

- Prevent and Create an Issue. Images created after this policy is enabled will be blocked from running if they don't meet the trust criteria, and a violation will also be recorded as a security issue. For images that were already running—and which don't meet the trust criteria—a posture issue is created.For the purposes of this documentation, we will refer to this as the **prevent** action.

  These issues block deployment in real-time.

  The prevent action is cloud-agnostic (supports AWS, Azure, GCP, OCI) and enforced at the Kubernetes cluster level via the admission controller.

Before you begin working with trusted image policies

Consider the following prerequisite:

- All trusted image policies are designed for runtime security enforcement. However, to ensure that policies with the prevent action can successfully block untrusted images, the policy's scope must be on a Kubernetes cluster where the admission controller is enabled.

Trusted image policy enforcement logic

The evaluation of trusted image policies is based on logic that determines the following critical outcomes: Which container images are trusted in a given scope and which violations result in the creation of security and posture issues.

- Utilizing an "OR-based" approach

  Cortex utilizes an "OR-based" approach, with no explicit priority or rule order when resolving policy conflicts.

  This enables Cortex to prioritize the most permissive result among conflicting policies with prevent actions, while enforcing the most restrictive result between policies with issue-only actions.

  - Prevent actions: Minimizing prevention, and allowing as many images as possible to be trusted, is desirable in order to avoid blocking workloads from running.
  - Issue creation: Minimizing the number of issues generated is desirable to avoid issue redundancy, unnecessary analysis, and issue handling.

  Understanding the approach and the logic behind it helps you create policies efficiently.

  Recommendations include:

  - When possible, the optimal way to define all trust criteria is to define the criteria in one policy.
  - If you are defining cluster-level trust criteria along with other options at the namespace/registry level, understand that because there are no priority/order options, you must plan your policies accordingly.

- Trusting and allowing images by scope

  Trusted image policies provide granular control by ensuring that policies only impact the specific environment (scope) for which they are defined. An image can be trusted in one scope and untrusted in another. For example, a third party image may be trusted in your Development environment but immediately untrusted in your Production environment.

  If you have not defined any trusted image policies for a specific scope, the default behavior is that all images within that scope are implicitly trusted However, once a policy is defined for a scope, any image not explicitly trusted by that policy is automatically untrusted.

  Policy enforcement is directly tied to the defined scope. When you define the Asset Group scope for a policy (for example, a specific cluster's Test namespace), the policy applies only to the resources within that exact definition. The more narrowly defined your scope, the more targeted the policy's enforcement will be, ensuring the enforcement does not affect unrelated resources.

  If multiple policies with prevent actions cover the same scope, the logic is additive (allow-wins). As long as at least one of those policies results in the image being trusted, the image will be allowed to run.

  > **NOTE:**
  >
  > Trusted image policies do not override other policies and their rules, such as malware policies, misconfiguration policies, and secrets policies.

- Generating issues

  The following points clarify the logic for generating runtime security issues and posture issues based on your defined policies:

  - Runtime security issues:

    A runtime security issue is generated only if an image is untrusted by all policies. If at least one policy trusts the image, no runtime security issue is created. Additionally, each policy with a prevent action that identifies an issue will generate only one corresponding runtime security issue.

  - Posture issues:

    For policies with the prevent action:

    - if an image is prevented from running, no posture issues are created by any policy within that scope.

    - Posture issues are created if images that are running violate any policy within the scope.

    For policies with the issue-only action, posture issues are created at the granular level of one issue per untrusted image per policy that fails the trust criteria. The posture Issues are created on the relevant asset type and are titled accordingly:

    - Kubernetes workload asset type: The issue title will be Untrusted image running in Kubernetes workload X.

    - Virtual machine asset type: The issue title will be Untrusted image running in Virtual Machine Y.

    For each issue regardless of asset type, the description will be either:

    - For untrusted images:

      [AssetType] [AssetName] is running the image {ImageName}. that does not comply with the trust criteria defined in the Trusted Images policy [PolicyName].

    - For cases when there's missing information:

      [AssetType] [AssetName] is running the image {ImageName}. Information is missing to determine compliance with the trust criteria defined in the Trusted Images policy [PolicyName].

    The issues will also include the image name, a link to the image asset page, the relevant Kubernetes hierarchy (cluster and namespace) for Kubernetes workloads only, and more.

- Policy evaluation: Handling unknown, missing or partial image data

Situations may arise where insufficient information is available at the time of deployment to conclusively arrive at a trust verdict. This means Cortex cannot confirm an image's trusted or untrusted status. Examples of such situations where there is a lack of complete image metadata include:

- Initial discovery or missing scan data: If an image is being evaluated for the first time, some vital information may be temporarily unavailable that would generally have been derived from a prior comprehensive scan, such as the image's Base Layer data or its CLI scan status.
- Partial deployment metadata: The trust criteria specified in your trusted image policy may include image metadata that is not present in the deployment file. For example, a policy might mandate a specific tag, but the Kubernetes deployment resource uses only the image digest (SHA) and omits the tag.

Cortex handles these situations differently based on the policy's action:

- For policies with issue-only actions: Cortex creates a posture issue and a runtime security issue, flagging the image as potentially untrusted due to insufficient data. The image is allowed to deploy, but the issue prompts the user to investigate the data gap.
- For policies with prevent actions: Cortex relies on the user-defined Action when trust verdict is unavailable value in the policy's action section.

> **CAUTION:**
>
> The default behavior for the prevent action when trust criteria are unavailable is Create an Issue. Be aware that changing this default will mean an incomplete deployment file can block a workload.

- Policy decision logic for workloads

  Situations might arise where within the same workload, multiple images or multiple policies yield conflicting trust results. When these conflicts occur—such as one policy trusting an image while another prevents it, or a single deployment containing both trusted and untrusted images—Cortex relies on the following defined logic to determine the final deployment outcome and what issues, if any, are generated.

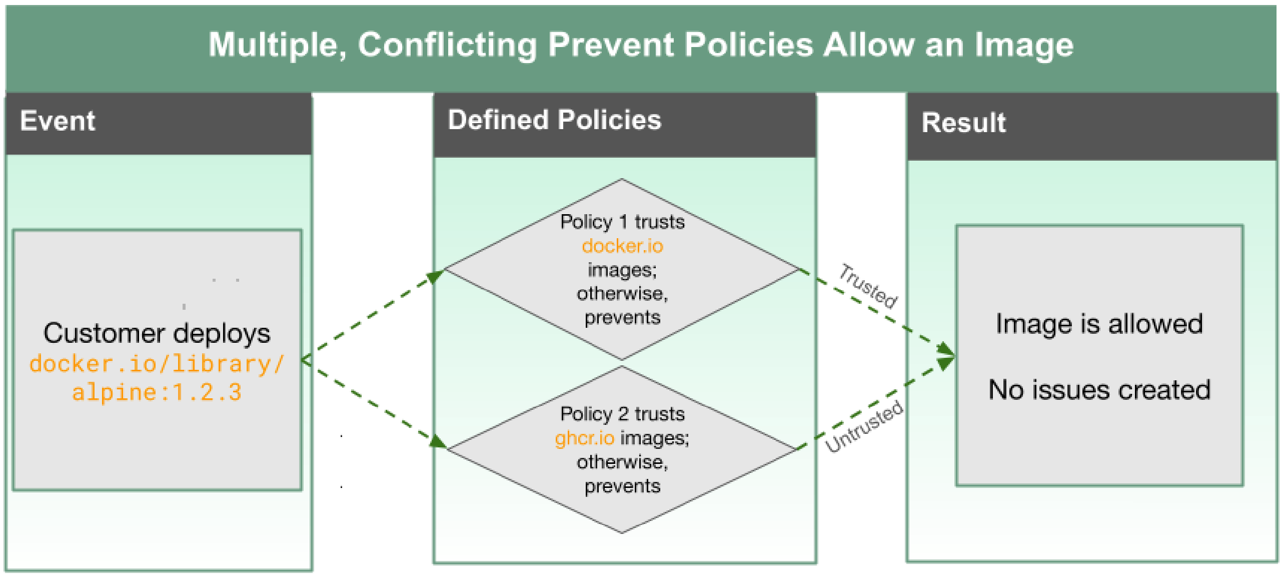
- For policies with issue-only actions: Any time an image fails the trust criteria for an Issue-only action policy, a Posture Issue is created. This action is non-blocking; the image is allowed to deploy regardless of the issue.
- For policies with prevent actions:
  - Conflicting allow vs. prevent policies: If multiple prevent action policies apply to a single image within a workload, and one policy allows the image while another does not trust it, the image is ultimately trusted and allowed to run. This follows the "Allow-Wins" principle.
  - Conflicting Images within the same workload: If a single workload (meaning, one deployment) contains both trusted and untrusted images, the entire deployment is blocked. The security risk posed by the single untrusted image prevents the entire workload from running.

Policy enforcement examples

This section provides examples that help illustrate the policy enforcement logic described above.

Example: Handling contradictory policies

This example illustrates how Cortex determines how to allow or prevent images, and create issues, when multiple policies conflict.


Multiple, Conflicting Prevent Policies Allow an Image
Event
Defined Policies
Result
Customer deploys docker.io/library/alpine:1.2.3
Policy 1 trusts docker.io images; otherwise, prevents
Policy 2 trusts ghcr.io images; otherwise, prevents
Trusted
Untrusted
Image is allowed
No issues created

1. User deploys the docker.io/library/alpine:1.2.3 image.
2. Trusted image policies are validated at set periodic scan intervals or when a resource is deployed.
3. Two trusted image policies with prevent actions evaluate the image to determine its trust status. Cortex uses an "OR-based" evaluation for conflicting prevent action policies to determine if the image is trusted.

4. Policy 1 allows the image because the image is in the **docker.io** registry. So even though Policy 2 does not trust the image, the trust verdict is trusted and image deployment proceeds.

5. No runtime security issues are created because the image is trusted by one policy—namely, Policy 1.

6. No posture issues are created because:

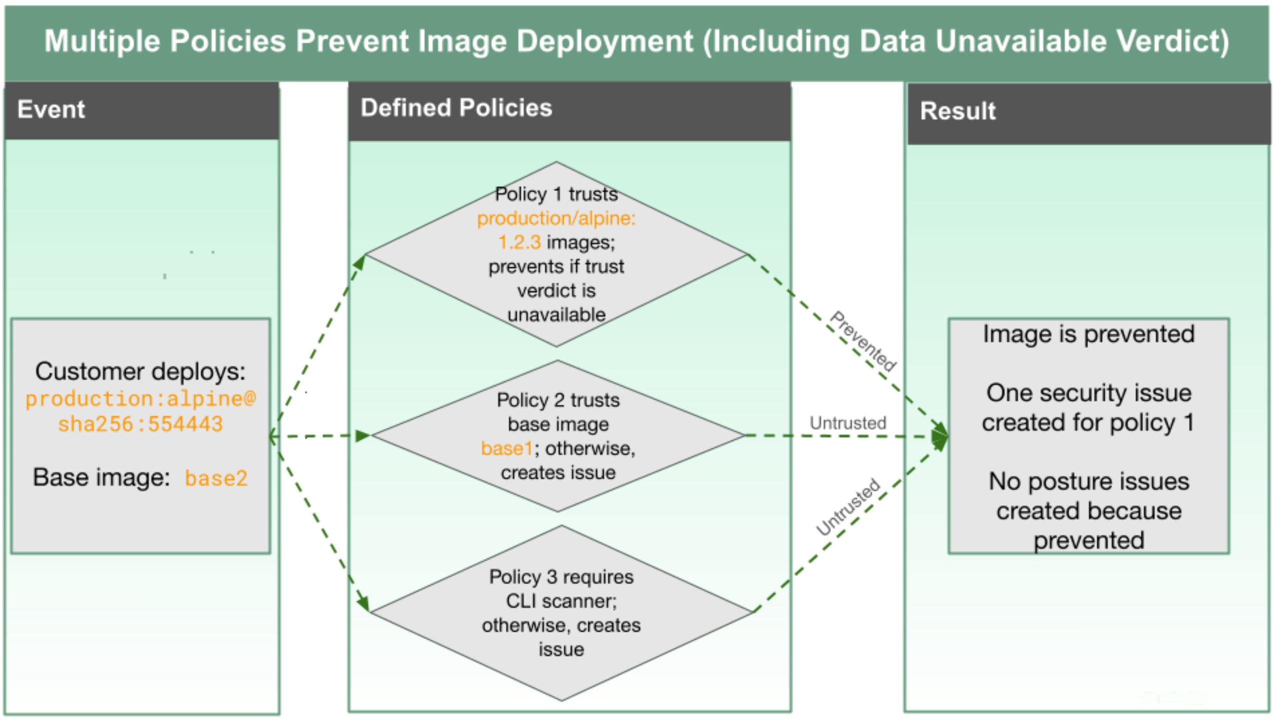
   - There are no policies with issue-only actions—only policies with prevent actions.

   - The user defined two trusted registries, and a trusted image comes from one of them.

     The user does not anticipate any further issues.

Example: Preventing images based on policies with prevent actions

This example illustrates how Cortex determines how images are prevented from deployment in cases where a data verdict is unavailable.


Multiple Policies Prevent Image Deployment (Including Data Unavailable Verdict)
Event
Defined Policies
Result
Customer deploys: production:alpine@sha256:554443
Base image: base2
Policy 1 trusts production/alpine:1.2.3 images; prevents if trust verdict is unavailable
Policy 2 trusts base image base1; otherwise, creates issue
Policy 3 requires CLI scanner; otherwise, creates issue
Prevented
Untrusted
Untrusted
Image is prevented
One security issue created for policy 1
No posture issues created because prevented

1. User deploys the **production:alpine@sha256:554443** image with the base image of **base2**.

2. Trusted image policies are validated at set periodic scan intervals or when a resource is deployed.

3. Three policies with prevent actions evaluate the image to determine its trust status. Cortex uses an "OR-based" evaluation for conflicting policies to determine if the image is trusted.

- Policy 1's criteria **production/alpine:1.2.3** only partially match the deployed image's full image identifier, **production:alpine@sha256:554443**. For example, the deployed image is missing the tag.

  If the image developer built the image, pushed it to the registry, and tagged it as **:1.2.3**, and that specific content generated the digest **sha256:554443...**, then they are a match at that moment in time. But this could change, so there is no definitive trust verdict.

  Because policy 1 has a prevent action with the additional **Action when trust verdict is unavailable** option set to **Prevent and create an issue**, the image is untrusted.

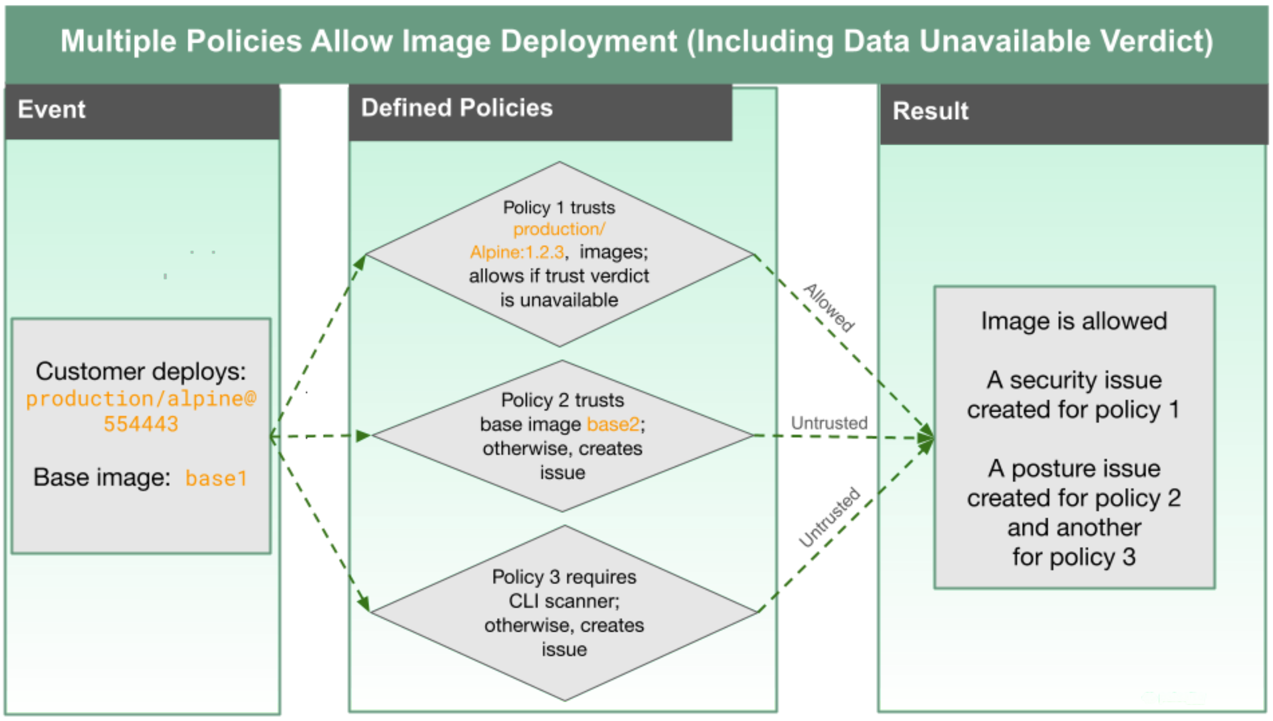
- Policy 2, with its issue-only action, only trusts **base1** images, so this policy considers the image untrusted.
- Policy 3, with its issue-only action, requires the image to have been scanned by a CLI scanner, so the image is untrusted.

4. Because all three policies do not trust the image, the image is prevented and deployment does not proceed.
5. A runtime security issue is created because of at least one policy does not trust the image. Only one runtime security issue is created, even though three policies don't trust the image.

   No posture issues are created because the image was prevented. Posture images must be actionable, and no actions can be taken on a prevented image.

Example: Allowing images based on policies with prevent actions

This example illustrates how Cortex determines how images are allowed to deploy in cases where a data verdict is unavailable.


Multiple Policies Allow Image Deployment (Including Data Unavailable Verdict)
Event
Defined Policies
Result
Customer deploys: production/alpine@ 554443
Base image: base1
Policy 1 trusts production/ Alpine:1.2.3, images; allows if trust verdict is unavailable
Policy 2 trusts base image base2; otherwise, creates issue
Policy 3 requires CLI scanner; otherwise, creates issue
Allowed
Untrusted
Untrusted
Image is allowed
A security issue created for policy 1
A posture issue created for policy 2 and another for policy 3

1. User deploys the production:alpine@sha256:554443 image with the base image of base2.
2. Trusted image policies are validated at set periodic scan intervals or when a resource is deployed.
3. Three policies evaluate the image to determine its trust status. Cortex uses an "OR-based" evaluation for conflicting policies to determine if the image is trusted.
    - Policy 1's criteria production/alpine:1.2.3 only partially match the deployed image's full image identifier, production:alpine@sha256:554443. For example, the image is missing the tag.

      If the image developer built the image, pushed it to the registry, and tagged it as :1.2.3, and that specific content generated the digest sha256:554443..., then they are a match at that moment in time. But this could change, so there is no definitive trust verdict.

      Because policy 1 has a prevent action with the additional Action when trust verdict is unavailable option set to Create an issue, the image is trusted.
    - Policy 2, with its issue-only action, only trusts base1 images, so the policy considers the image is untrusted.
    - Policy 3, with its issue-only action, requires the image to have been scanned by a CLI scanner, so the image is considered untrusted.
4. The image is allowed and deployment proceeds, because one of the three policies trusts the image and Cortex uses an "OR-based" approach.
5. No runtime security issue is created because of the trust verdict is to trust and allow.

   No posture issues are created because the image was prevented. Posture images must be actionable, and no actions can be taken on a prevented image.

8.2.2 | Cloud Workload Policies page

The Cloud Workload Policies page allows users to manage policies that define security and compliance actions for cloud workloads. Users can create, edit, filter, and manage policies through a structured table and widget panel.

> **NOTE:**
>
> Keep the following caveats in my mind when working with Policies:

- Instance Administrators are able to view all facets of policies without restrictions, even if Scope Based Access Control (SBAC) roles are in effect. Learn more about SBAC.
- If you've been assigned a custom role with View/Edit permissions limited by SBAC, you may not be able to view certain policies.
- You can further narrow your search on the Inventory page by using SBAC to limit the scope of the finding, issue, and case counts.

The Cloud Workload Policies page displays all the configured policies with the following fields.

Policy table columns

Field	**Description**
Policy Type	Defines the policy category: Misconfigurations, Secrets, Malware, Trusted Images.
Policy Name	The user-defined name of the policy.
Action	Defines the action taken when conditions match: Create an Issue (logs an issue) or Prevent and Create an Issue (prevents the action and logs an issue).
Severity	The severity level of the issue created: Critical, High, Medium, Low, or Informational.
Asset Groups	Predefined groups of assets to which the policy applies.
Open Issues	The number of unresolved issues associated with the policy.
Conditions	Define the detection rule by specifying the criteria that match relevant malware, secret, or trusted image findings.
Exceptions	Defines the exclusion criteria to omit malware, secret, or trusted image findings that meet specific conditions you want to exclude from the policy.

Field	Description
Evaluation Stage	Indicates at which stage in the SDLC the policy is evaluated.
Description	Additional details about the policy.
Created By	The user who created the policy.
Last Modified	The timestamp of the last modification.

8.2.2.1 | Widgets panel

The Cloud Workload Policies page includes a widget panel that provides a visual summary of policies:

- Policies by Type: Displays policies categorized as misconfiguration, secret, trusted images, or malware.
- Policies by Evaluation Stage: Shows the distribution of policies based on SDLC evaluation stages: Runtime, Deploy, or CI.

8.2.2.1.1 | Show or hide the widget panel

The widget panel provides a visual summary of policies based on policy type or evaluation stage.

To hide the widget panel, do the following:

1. Navigate to Posture Management → Rules & Policies → Policies → Cloud Workload.
2. On the Cloud Workload Policies page, click the Widget Panel icon at the top of the page.
3. The panel toggles between visible and hidden states.

8.2.2.2 | Change the layout of the policies table

1. Navigate to Posture Management → Rules & Policies → Policies → Cloud Workload.
2. In the Cloud Workload Policies page, click the More Options icon (⋮).
3. In the Layout tab, do the following:

- To remove columns, go to the In View section and search for a specific column. Click - next to the column to remove it from the table.
- To reorder columns, go to the In View section. Click and drag columns up or down to rearrange the columns.
- To add new columns, go to the Add Columns section. Click + next to the columns to include them in the table.

4. The table layout updates automatically based on your selections.

8.2.2.3 | Policy Details Panel

The policy details panel is displayed when you click a policy in the policy table. To view details of a cloud workload policy:

1. Navigate to Posture Management → Rules & Policies → Policies → Cloud Workload.
2. In the Cloud Workload Policies page, select the policy you want to check.

The policy panel displays the following details related to the selected policy:

- Policy details.
- Related rule settings.
- The number of issues opened as part of the policy. You can click on the link to navigate to the Issues and Cases section to check the issue details.

From the policy detail panel, you can:

- Enable or disable the policy
- Edit the policy
- Save as new
- Delete the policy

8.2.3 | Enable or disable a Cloud Workload Policy

1. Navigate to Posture Management → Rules & Policies → Policies → Cloud Workload.
2. In the Cloud Workload Policies page, click on the policy you want to enable or disable.
3. In the Details page, click the toggle button at the top to enable or disable the policy.

8.2.4 | Create a Cloud Workload Policy

You can create policies to address specific types of security risks or compliance requirements.

To create a cloud workload policy:

Navigate to **Posture Management** → **Rules & Policies** → **Policies** → **Cloud Workload**.

In the Cloud Workload Policy page, click Create Policy and select the type of policy you want to create:

Misconfiguration Policy

1. Enter a unique name and description. Note that these are mandatory fields.
2. The Evaluation Stage will be selected as Runtime.

   > **NOTE:**
   >
   > The Evaluation stage for Misconfiguration policies is supported only in the Runtime SDLC stage and is enforceable through the Kubernetes Admission Controller for clusters on-boarded using the Posture Management (KSPM) Connector.

3. Click Next
4. The **Summary** section on the right displays a real-time, interactive view of all policy configurations as users progress through the wizard. It **automatically updates** to reflect the current selections and settings, enabling seamless navigation between fields from any step in the wizard. It includes the following sections:
   - **General** – Policy name and description.
   - **Rules** – No. of selected rules and the asset types relevant to the selected rules.
   - **Scope** - The defined scopes included in the policy and SDLC stage.
5. In the Rules section:

- Click on Add Rules to select the rules that identify the violations that you want to track.

  A new window opens, displaying a list of all the existing predefined OOTB rules. See Rules filters for details on applying filters to refine the list of rules.

  > **NOTE:**
  >
  > Use Create a new Custom Detection Rule to define and add new custom rules as required.

- After completing your selection, click Select to confirm. The chosen rules are displayed in the Rules section, where you can toggle between the Cards and Grid view to display the rules in your preferred layout.

- In the Rules section, you can select one or more rules to modify the Severity, Policy Action and Remediation values, either individually or in bulk.

  > **NOTE:**
  >
  > Each rule may support different actions. While some include both Create an Issue and Prevent and Create an Issue, others provide only the Create an Issue option.

6. In the Scope section, for the Scope Selection Method , select Asset Groups or Default Asset Scopes, depending on your preference.

   1. If you select Asset Groups, you can choose between the following options:

   Select existing Asset groups

   The displayed list shows all Asset Groups available in the Runtime SDLC stageYou can select the asset group to which you want this policy to apply.

   Click Preview Selection to view all relevant Compute Assets associated with the selected asset groups.

   > **NOTE:**
   >
   > All non-relevant (non-compute) assets are automatically excluded from the included asset list.

   Add Group

   Click on Add Group. A new window opens to create a new Compute Asset group.

   - Enter a unique Group name and Description.
   - The displayed list of **Compute Assets** in the table is **pre-filtered** to show only the **relevant assets** based on the **rules selected in the previous section**. The **asset list is dynamically updated** and restricted to the applicable asset types of the selected rules, ensuring that users can select only valid and compatible assets for the new Compute Asset Group.

     You can filter these assets using the Show filter Panel button based on the fields Asset ID, Name, Provider and more.
   - When only an **asset filter** is defined, the system creates a **dynamic asset group**, that automatically includes assets that meet the specified filter criteria.

     When **specific assets are manually selected** from the asset list, a **static asset group is created**, containing only the explicitly chosen assets.

   2. On selecting Default Asset Scopes, you can further select the Assets Scope from the predefined **Asset Scopes** that are filtered based on the **selected rules in the previous section** and their applicable **Asset Types**. The Scope options are dynamically updated and limited to the applicable asset types of the selected rules, ensuring that users can select only valid and compatible scopes.

7. Click Done to complete the process and create the new Misconfiguration Workload Policy.

Malware Policy

1. Enter a unique name and description. Note that these are mandatory fields.
2. Select an SDLC Evaluation Stage. The following options are available.
   - CI
   - Runtime
   - Deploy
3. Click Next.
4. The **Summary** section on the right provides a real-time, readable view of all policy configurations as users progress through the wizard. It automatically updates to reflect current selections and settings. It includes the following sections:
   - **General** – Policy name and description.
   - **Conditions** – Selected rule filter and exclusion criteria.
   - **Scope** - The defined scopes included in the policy and SDLC stage.
   - Actions - Selected action type.
5. Configure the settings specific to the evaluation stage you select.

CI

- In the Conditions section, define the detection rule by specifying the criteria to identify relevant malware findings. You may also include exclusion criteria to filter out any malware findings that meet specific conditions you wish to exclude from this policy.
- Click Next.
- In the Scope section, select the checkbox to confirm that this selection applies the policy and its detection rules to All Cloud Workload Build Container Image asset types available at the CI SDLC stage.
- Click Next.
- In the Action section:
    - For Select an action, choose the option to Create an issue to log an issue if the policy is violated or Prevent and create an issue to prevent and create an issue.

      > **NOTE:**
      >
      > See Cloud Workload Preventive Action, to learn more about the Prevent action behavior and prerequisites.
      >
      > If the Prevent and create an issue action is selected, the preventive actions in the CI pipeline will trigger a Fail Pipeline by returning an exit code of 2 in the CI tool.

    - Under Issue Severity, choose **Critical, High, Medium or Low** to define the issue severity.
    - In the Remediation Guidance field, enter the optional remediation instructions.
- Click Done to complete the process and create the new Cloud Malware Workload Policy.

Runtime

- In the Conditions section, define the detection rule by specifying the criteria to identify relevant malware findings. You may also include exclusion criteria to filter out any malware findings that meet specific conditions you wish to exclude from this policy.
- Click Next.
- In the Scope section, for the Scope Selection Method, select Asset Groups or Default Asset Scopes, depending on your requirement.

- If you select Asset Groups, you can choose between the following options:

  Select existing Asset groups

  The displayed list contains all the asset groups for the Cloud Workload Container Images, Container Instances, Hosts (VM Instances), Serverless Functions or Kubernetes Workloads asset types that are available at the Runtime SDLC stage.

  Add Group

  - Click on Add Group. A new window opens to create a new **Compute Asset group**.
  - The displayed list of **Compute Assets** in the table is **pre-filtered** to show only only the **Compute asset types** as Cloud Workload Container Images, Container Instances, Hosts (VM Instances), Serverless Functions or Kubernetes Workloads asset types, ensuring that users can select only valid and compatible assets for the new Compute Asset Group.

    You can filter these assets using the Show filter Panel button based on the fields Asset ID, Name, Provider and more.
  - When only an **asset filter** is defined, the system creates a **dynamic asset group**, that automatically includes assets that meet the specified filter criteria.

    When **specific assets are manually selected** from the asset list, a **static asset group is created**, containing only the explicitly chosen assets.

  You can then select the asset group to which you want this policy to apply.
- On selecting Default Asset Scopes, you can further select the Asset Scopes as:
  - All Cloud Workload Assets)

    > **NOTE:**
    >
    > **Choosing this option results in the automatic selection of all other asset scopes in the list.**
  - All Cloud Workload Hosts(VM Instances)
  - All Cloud Workload Container Images
  - All Cloud Workload Container Instances
  - All Cloud Workload Kubernetes Workloads
  - All Cloud Workload Serverless Functions

- Click Next.

- In the Action section:
    - For Select an Action, choose the option to Create an issue to log an issue if the policy is violated or Prevent and create an issue to prevent and create an issue.

      > **NOTE:**
      >
      > See Cloud Workload Preventive Action, to learn more about the Prevent action behavior and prerequisites.

    - Under Issue Severity, choose **Critical, High, Medium or Low** to define the issue severity.
    - In the Remediation Guidance field, enter the optional remediation instructions.
- Click Done to complete the process and create the new Cloud Malware Workload Policy.

Deploy

- In the Conditions section, define the detection rule by specifying the criteria to identify relevant malware findings. You may also include exclusion criteria to filter out any malware findings that meet specific conditions you wish to exclude from this policy.
- Click Next.
- In the Scope section, for the Scope Selection Method, select Registry Images in Cloud Workload Asset Groups or the All Cloud Workload Registry Images, depending on your requirement.
    - If you select Registry Images in Cloud Workload Asset Groups ,this policy applies only to Cloud Workload Container Registry Images asset types in those groups that are available at the Deploy SDLC stage. A list of all these available asset groups is displayed. You can then select the asset group to which you want this policy to apply.
    - On selecting All Cloud Workload Registry Images, the policy and its detection rules will apply to All Cloud Workload Container Registry Images asset types available at Deploy SDLC Stage.
- Click Next.
- In the Action section:
    - For Select an Action, the default option is Create an issue to log an issue if the policy is violated.
    - Under Issue Severity, choose Critical, High, Medium or Low to define the issue severity.
    - In the Remediation Guidance field, enter the optional remediation instructions.
- Click Done to complete the process and create the new Cloud Malware Workload Policy.

Secret Policy

- Enter a unique name and description. Note that these are mandatory fields.
- Select an SDLC Evaluation Stage. The following options are available.
    - CI
    - Runtime
    - Deploy

- Click Next.
- The Summary section on the right provides a real-time, readable view of all policy configurations as users progress through the wizard. It automatically updates to reflect current selections and settings. It includes the following sections:
    - General – Policy name and description.
    - Conditions – Selected rule filter and exclusion criteria.
    - Scope - The defined scopes included in the policy and SDLC stage.
    - Actions - Selected action type.
- Configure the settings specific to the evaluation stage you select.

  CI

    - In the Conditions section, define the detection rule by specifying the criteria to identify relevant malware findings. You may also include exclusion criteria to filter out any malware findings that meet specific conditions you wish to exclude from this policy.
    - Click Next.
    - In the Scope section, select the checkbox to confirm that this selection applies the policy and its detection rules to All Cloud Workload Build Container Images asset types available at the CI SDLC stage.
    - Click Next.
    - In the Action section:
        - For Select an action, choose the option to Create an issue to log an issue if the policy is violated or Prevent and create an issue to prevent and create an issue.

          > **NOTE:**
          >
          > See Cloud Workload Preventive Action, to learn more about the Prevent action behavior and prerequisites.

        - Under Issue Severity, choose **Critical, High, Medium or Low** to define the issue severity.
        - In the Remediation Guidance field, enter the optional remediation instructions.
    - Click Done to complete the process and create the new Cloud Malware Workload Policy.

  Runtime

- In the Conditions section, define the detection rule by specifying the criteria to identify relevant malware findings. You may also include exclusion criteria to filter out any malware findings that meet specific conditions you wish to exclude from this policy.
- Click Next.
- In the Scope section, for the Scope Selection Method, select Asset Groups or Default Asset Scopes, depending on your requirement.

- If you select Asset Groups, you can choose between the following options:

  Select existing Asset groups

  The displayed list contains all the asset groups for the **Cloud Workload Container Images, Container Instances, Hosts (VM Instances), Serverless Functions or Kubernetes Workloads** asset types that are available at the Runtime SDLC stage.

  Add Group

  - Click on Add Group. A new window opens to create a new Compute Asset group.
  - The displayed list contains all the asset groups for only the **Compute asset types** as **Cloud Workload Container Images, Container Instances, Hosts (VM Instances), Serverless Functions or Kubernetes Workloads**, ensuring that users can select only valid and compatible assets for the new Compute Asset Group.

    You can filter these assets using the Show filter Panel button based on the fields Asset ID, Name, Provider and more.
  - When only an asset filter is defined, the system creates a **dynamic asset group** that automatically includes assets that meet the specified filter criteria.

    When **specific assets are manually selected** from the asset list, **a static asset group is created,** containing only the explicitly chosen assets.

  You can then select the asset group to which you want this policy to apply.
- On selecting Default Asset Scope, you can further select the Asset Scopes as
  - All Cloud Workload Assets

    > **NOTE:**
    >
    > **Choosing this option results in the automatic selection of all other asset scopes in the list.**
  - All Cloud Workload Hosts(VM Instances)
  - All Cloud Workload Container Images
  - All Cloud Workload Container Instances
  - All Cloud Workload Kubernetes Workloads
  - All Cloud Workload Serverless Functions
- Click Next.

- In the Action section:
    - For Select an action, choose the option to Create an issue to log an issue if the policy is violated or Prevent and create an issue to prevent and create an issue.

      > **NOTE:**
      >
      > See Cloud Workload Preventive Action, to learn more about the Prevent action behavior and prerequisites.

    - Under Issue Severity, choose **Critical, High, Medium or Low** to define the issue severity.
    - In the Remediation Guidance field, enter the optional remediation instructions.
- Click Done to complete the process and create the new Cloud Malware Workload Policy.

Deploy

- In the Conditions section, define the detection rule by specifying the criteria to identify relevant malware findings. You may also include exclusion criteria to filter out any malware findings that meet specific conditions you wish to exclude from this policy.
- Click Next.
- In the Scope section, for the Scope Selection Method select Registry Images in Cloud Workload Asset Groups or the All Cloud Workload Registry Images, depending on your requirement.
  - If you select Registry Images in Cloud Workload Asset Groups ,this policy applies only to Cloud Workload Container Registry Images asset types in those groups that are available at the Deploy SDLC stage. A list of all these available asset groups is displayed. You can then select the asset group to which you want this policy to apply.
  - On selecting All Cloud Workload Registry Images, the policy and its detection rules will apply to All Cloud Workload Container Registry Images asset types available at Deploy SDLC Stage.
- Click Next.
- In the Action section:
  - For Select an Action, the default option is Create an issue to log an issue if the policy is violated.
  - Under Issue Severity, choose Critical, High, Medium or Low to define the issue severity.
  - In the Remediation Guidance field, enter the optional remediation instructions.
- Click Done to complete the process and create the new Cloud Malware Workload Policy.

Trusted Images

1. Enter a unique name and description. Note that these are mandatory fields. The SDLC Evaluation Stage is preset to Runtime.
2. Click Next.
3. Configure the policy's condition settings.
   a. In the Conditions section, specify the criteria to identify relevant images.

      You can specify criteria to define both broad policies and strict policies, for example: `Trust images (from registryX or registryY) OR (digestA or digestB)`. An example of criteria for a broad policy could be `all images from gcr.io/myorg/` while an example of criteria for a strict policy could be: `gcr.io/myorg/app@sha256:abc123`.

You can also include exclusion criteria to filter out any images that meet specific conditions for exclusion from this policy.

Because trust is subjective, context-dependent, and scope-based, we recommend you create finely-tuned criteria. For example, an image might be trusted in a low-risk demo environment because it has relaxed patching requirements, but it would be instantly blocked as untrusted in a production environment. For more information, see Trusted Image Policies.

Considerations for specifying criteria for the policy's conditions

- If you include a base image as a criterion in your trusted image policy, ensure that the base image itself is successfully scanned and pre-ingested into the system.
- Do not use Scanned by CLI = Yes as the sole criterion for establishing image trust. The CLI scanning status is generally used as an indicator that contributes to overall trustworthiness. Instead, combine the CLI scanning status with stronger, verifiable identifiers like the image registry, signature, and/or base image.
- Define trust criteria only using metadata that is consistently and explicitly included in your deployment files. The policy cannot establish trust if the required metadata (for example, a specific tag or label) is missing from the image definition.
- Avoid using mutable tags as criteria for establishing image trust. Because the underlying image associated with a mutable tag can change without warning, basing trust on it can lead to unexpected outcomes. We strongly recommend enforcing immutable tags—a hallmark of secure, mature CI/CD pipelines—which is supported by all major registries (such as Docker Hub, ACR, ECR, GAR).
- When using trust criteria that rely on image metadata, such as the base layer information or specific tags, we recommend that you pre-ingest the images into Cortex Cloud. While it is possible to base trust on an image which has not been pre-ingested, omitting this step can significantly impact the performance of your policy evaluation, which can slow down critical CI/CD workflows.

b. Click Next.

4. Configure the policy's scope settings.

   a. In the Scope section, for the Scope Selection Method select Asset Groups or Default Asset Scopes.

        - Asset Groups. The policy applies only to Cloud workload container images, container instances, hosts (VM instances), or Kubernetes workload asset types in those groups that are available at the Runtime SDLC stage. A list of available asset groups is displayed. You can then select the asset group to which you want this policy to apply.

          We recommend narrowing the asset group scope to ensure that a policy only checks relevant assets. For more information, see Trusted Images Policies.

          Consider the following when specifying criteria for the policy's scope:

            - Exclude system-critical Kubernetes namespaces, such as kube-system, from the policy scope to avoid interfering with core cluster operations.
            - If you select an asset group that contains a specific namespace, the policy will apply only to resources in that namespace—not the entire cluster.

        - Default Asset Scopes. On selecting Default Asset Scopes , you can further select the Asset Types:

            - All Cloud Workload Assets
            - All Cloud Workload Container Images
            - All Cloud Workload Kubernetes Workloads
            - All Cloud Workload Hosts (VM Instances)

    b. Click Next.

5. Configure the policy's action settings. In the Action section:

    a. For Select an Action, choose either Create an issue to log an issue if the policy is violated or Prevent and create an issue to prevent and create an issue. For more information, see Preventative action.

    b. If you select Prevent and create an issue as the policy's action, an additional Action when trust verdict is unavailable option becomes available. This is for situations where there is insufficient information available for determining if the image is trusted. The default is Prevent and create an issue.

    c. Under Issue Severity, choose Critical, High, Medium, or Low to define the issue severity.

    d. In the Remediation Guidance field, enter optional remediation instructions.

   Issues are automatically closed when the affected asset is removed from the inventory or when the policy is deleted. You can manually close issues at any time.

6. Click Done to complete the process and create the new policy.

8.2.5 | Use an existing policy to create a new Cloud Workload policy

1. Navigate to Posture Management → Rules & Policies → Policies → Cloud Workload.
2. In the Cloud Workload Policies page, click the policy you want to enable or disable.
3. In the Details page, click the More Options icon (⋮) and then select Save as new.
4. Modify the necessary fields in the Policy Name, Conditions, Scope, and Actions screens.
5. Click Done to create the new custom policy.

8.2.6 | Edit a Cloud Workload Policy

1. Navigate to Posture Management → Rules & Policies → Policies → Cloud Workload.
2. In the Cloud Workload Policies page, click the policy you want to enable or disable.
3. In the Details page, click the Edit icon.
4. Make the necessary changes.
5. Click Done to save the changes.

8.2.7 | Delete a Cloud Workload Policy

1. Navigate to Posture Management → Rules & Policies → Policies → Cloud Workload.
2. In the Cloud Workload Policies page, click the policy you want to delete.
3. In the Details page, click the More Options icon (⋮) and then select Delete policy.
4. Click Delete to confirm.

8.2.8 | Cloud Workload Preventive Action

Some Cloud Workload policies provide a Prevent and Create an Issue action that enforces compliance during deployments.

Prevention action for Runtime stage Policies

The Prevent action at Runtime applies only to Kubernetes Workload Images assets.

When a Kubernetes Workload image violates a policy, the Kubernetes Admission Controller (on clusters where the KSPM Connector is deployed and Admission Control is enabled) can block it from being admitted to the cluster.

For all other asset types within the policy scope, no runtime prevention will occur. Instead, the violation will result in an Issue being created.

Prerequisites

Ensure that your cluster has the Posture Management (KSPM) Connector deployed with the Admission Controller functionality enabled.

You can manage these deployments from the Kubernetes Connectivity Management page.

To access the Kubernetes Connectivity Management, navigate to the following URL in your tenant environment: https://[TENANT-ADDRESS]/cwp/k8s-management.

Important considerations

- Recommended Approach: Begin with the Create an Issue action to validate results before selecting Prevent and Create an Issue. This helps prevent potential disruptions to your applications or development workflows.
- Impact on New Deployments: The Prevent and Create an Issue action affects only new or future deployments that meet the prevention criteria. It does not impact cloud workload assets that are already deployed.

Prevention action for CI stage Policies

Prevention actions in the CI stage triggers a pipeline failure by returning an exit code of 2 in the CI tool.

8.3 | Cloud Workload Rules

Rules: Cloud Workload Rules define the criteria for identifying security violations. This criteria can be applied to assets in your cloud environment and to findings generated by Cortex Cloud.

Rules only enable the detection of security violations. They must be included in a policy to trigger a preventive response or generate an alert in the form of an issue.

8.3.1 | Default (pre-defined) Rules

Cortex Cloud includes a number of pre-defined rules to secure your cloud runtimes. These rules are used by default policies to prevent security violations and create issues.

8.3.2 | Custom (user-defined) Rules

Custom Rules or Custom Detection Rules allow you to define and implement tailored security and compliance checks within cloud workloads. These rules enable organizations to detect specific conditions, vulnerabilities, or misconfigurations that might not be covered by built-in system rules.

A custom rule consists of the following components:

- Scanner: Defines the mechanism by which the rule inspects the cloud assets. You need to select the scanner type that will implement the rule. Every time the selected scanner runs, all the rules associated with that scanner are also executed. The available scanner types are:
  - Agentless Disk Scan: Rules that use Agentless Disk Scanner to inspect the container images on which the Agentless scanner runs. You can specify different rules for containers running different OSes. For example, you can create a rule that checks for incorrect or malicious entries in the etc\hosts file on Windows images.
  - Kubernetes Connector: Rules that use the Kubernetes Connector scanner to inspect Kubernetes environment variables and resources such as Namespaces, ReplicaSets, Deployments and more.
  - XDR Agent: Rules that use XDR Agent Scanner to perform custom compliance checks by executing user-defined Python scripts, offering a tailored approach to compliance validation.
- Rule (Condition): Defines the detection criteria. This is specified as Rego or Python statements that evaluate assets, findings, and their associated attributes to identify security violations based on the selected scanner.
- Severity: The selected value is included in issues that are created as a result of rule violation.
- Compliance Controls: Associates the custom rule with a custom compliance control. If the rule detects the security violation, it will invoke the corresponding compliance control, thereby including the violation in relevant compliance reports.

8.3.3 | Cloud Workload Rules page

The Cloud Workload Rules page allows users to manage rules. Users can create, edit, filter, and manage rules.

NOTE:

Keep the following caveats in my mind when working with Rules:

- Instance Administrators are able to view all facets of Rules without restrictions, even if Scope Based Access Control (SBAC) roles are in effect. Learn more about SBAC.
- If you’ve been assigned a custom role with View/Edit permissions limited by SBAC, you may not be able to view specific Rules.
- You can further narrow your search in a Rules table by using SBAC to limit the scope of the finding, issues, and case counts.

The Widget section enables the users to get 'at-a-glance' based on Platform, Rule type and Scanner type.

The Cloud Workload Rules page displays both the default rules and user-configured rules, with the following fields.

Rule table columns

Column Name	**Description**
Rule ID	A unique identifier assigned to each rule.
Rule Name	The name of the rule, typically defined by the user or system.
Description	A brief summary of the rule's purpose and functionality.
Policies	Lists the policies in which the rule is included.
Controls	Compliance controls associated with the rule for regulatory adherence.
Platform	Specifies the platform or environment the rule applies to. For example: Linux, Windows or Kubernetes.
Scanner	The tool or method used to evaluate findings, such as *Inventory Scanner*, *Agentless Disk Scan, Host Scanner, Kubernetes Connector or Kubernetes File System Scanner* .
Severity	Defines the severity of the rule.
Data Type	The type of data the rule evaluates. For example: Hosts or Kubernetes Resources
Created By	The user who created the rule.

Column Name	Description
Last Modified	The date and time the rule was last updated.
Rule Type	Indicates whether the rule is a **Built-in** or Custom rule.
Remediation	Defines the remediation steps to address the detected misconfiguration.
Rule applicable assets	Supported applicable asset types.
Rule available actions	Indicates whether the available action is **Prevent and Create an Issue** or **Create an Issue**
Rule standards	Associated compliance standards or controls
Open issue	No. of open issue related to this rule.

8.3.3.1 | Filter page results

You can use Show filter Panel button in the upper-right corner of the Rules page to filter the existing rules based on different filter criteria, as described below:

Table 1. Rule Filter table

Filter	Allowed Values
Rule Name	Rule names and empty values
Description	Rule description and empty values

Filter	Allowed Values
Policies	No. of policies
Controls	No. of controls
Platform	*Linux, Windows and Kubernetes*
Scanner	*Agentless Disk Scan, Host Scanner, Kubernetes Connector, Kubernetes File System Scanner* and *Inventory Scanner*
Data type	*Hosts, Kubernetes Resources*
Severity	*Informational, Low, Medium, High* and *Critical*
Created by	System or specific username
Last modified	Selected date and time
Rule type	*Built-in* and *Custom*
Remediation	Remediation values
Rule applicable assets	Supported applicable asset types.
Rule available actions	*Prevent and Create an Issue* and *Create an Issue*

Filter	Allowed Values
Rule standards	Associated compliance standards or controls
Open Issues	No. of open issue.
Rule ID	Unique Id of a rule.

8.3.3.2 | Change the layout of the rules table

1. Navigate to **Posture Management > Rules > Cloud Workload**.
2. In the **Cloud Workload Rules** page, click the **More Options** icon (⋮).
3. In the **Layout** tab, do the following:
    - To add or remove columns, search for a specific column and:
        - Click **+** to add it to the table.
        - Click **-** to remove it from the table.
    - To reorder columns, go to the **In View** section and **click and drag** columns up or down.
    - To add new columns, go to the **Add Columns** section and click **+** to include them in the table.
4. The table layout updates automatically based on your selections.

8.3.3.3 | Rule details panel

The rule panel displays the following details related to the selected rule:

- Details of the rule like scanner details, remediation details and more.
- Compliance Controls for the rule.

This panel enables you to:

- Edit the rule
- Save as new
- Delete the rule

8.3.4 | Create a new Custom Detection Rule

Abstract

Create Custom Detection Rules to check your organization’s assets.

Creating Custom Detection Rules give you the flexibility to define and enforce security best practices tailored to your organization's objectives, as well as regulatory requirements not already covered by the compliance standards in our catalog.

Before you begin

Ensure you have a custom compliance control defined to associate the Custom Detection Rule to. For more information, see Use a built-in or custom standard.

How to create a Custom Detection Rule

1. Go to Posture Management → Rules & Policies → Rules → Cloud Workload.
2. In the Cloud Workload Rules page, click Create Custom Rule.
3. Enter the following settings:
   - Rule name: A descriptive name for the custom rule.
   - Description: An optional field for adding additional details or context about the rule, such as its purpose or intended behavior.
4. Select a Scanner to execute the Custom Detection Rule and its associated script. The options are:
   - Agentless Disk Scan
   - Kubernetes Connector
   - XDR Agent
5. Configure settings specific to the scanner you select.

   Agentless Disk Scan settings

Field	Description
Operating System	The operating system targeted by the rule. The available options are: • Linux • Windows
Input file(s) path	The full file path for one or more files. For example, `/nfs/an/disks/jj/home/dir/file.txt`
Define the Rule (Rego)	Use Rego to define the custom detection logic. Use the default code in this box as a reference or starting point. Click read here for more information how to use Rego syntax.

Kubernetes Connector settings

Field	Description
Kubernetes Resources	From the drop down, select one or more from the following:

Field	Description
	• Namespaces: Logical partitions within a Kubernetes cluster that allow resource isolation and organization. • ReplicaSets: Ensures a specified number of pod replicas are running at all times by automatically scaling up or down. • Deployments: Manages and control pod replicas by providing declarative updates for ReplicaSets, enabling rolling updates and rollbacks. • StatefulSets: Deploys stateful applications that require persistent identity and storage, ensuring stable pod names and ordered scaling. • DaemonSets: Ensures that a copy of a specific pod runs on all or selected nodes in the cluster, commonly used for logging and monitoring agents. • Jobs: Runs one-time or short-lived workloads that complete execution and then terminate. • CronJobs: Defines scheduled jobs that run at specified times or intervals, similar to Linux cron jobs. • ClusterRoles: Defines permissions at the cluster level, granting access to resources across all namespaces. • Roles: Defines permissions within a specific namespace, restricting access to resources within that namespace. • RoleBindings: Associates a role with a user, group, or service account within a specific namespace. • ClusterRoleBindings: Associates a cluster role with users, groups, or service accounts at the cluster-wide level. • NetworkPolicies: Defines rules that control the communication between pods and other network entities within the cluster, enforcing security restrictions. • Services: Exposes a set of pods as a network service, allowing stable communication within and outside the cluster. • ServiceAccounts: Provides an identity for pods to authenticate against the Kubernetes API, allowing controlled access to resources.

Field	Description
	• Endpoints: Represents the actual network addresses of the pods backing a service, dynamically updated as pods start or stop. • Ingresses: Manages external access to services, providing HTTP/HTTPS routing, load balancing, and SSL termination. • ConfigMaps: Stores non-sensitive configuration data in key-value pairs, allowing applications to retrieve configuration without modifying container images. • Secrets: Securely stores and manage sensitive data, such as API keys, passwords, and certificates, in an encrypted format. • Nodes: Defines the physical or virtual machines that run the workloads in a Kubernetes cluster.
Define the Rule (Rego)	Use Rego to define the custom detection logic. Use the default code in this box as a reference or starting point. Click read here for more information how to use Rego syntax.

XDR Agent settings

Field	Description
Custom Code Execution	Enable this setting for the scanner to perform custom compliance checks by executing user-defined Python scripts. **NOTE:** Only users with the following roles can enable or disable Custom Code Execution: • Account Admin • Instance Administrator • Deployment Admin • Privileged Security Admin Click Confirm to accept the following terms: • The Python scripts you provide will be executed in your cloud environment(s). • This capability is solely for the purpose of enabling you to define the compliance check rules for your cloud environment(s). Any other purposes are expressly prohibited. • Any actions involving WRITE, MODIFY, or DELETE operations of your cloud environment(s) are strictly prohibited. It is your responsibility to ensure that your custom Python scripts only perform read-only operations of your cloud environment(s) explicitly for compliance check purposes. • You are solely responsible for the quality, content, use, and execution results of your Python script. You assume all risks and liabilities arising from executing your Python script(s), including any potential errors, damages, or consequences resulting from its use. After you confirm accepting the terms, the rest of the XDR Agent settings appear.
Operating System	The operating system targeted by the rule. The available options are: • Linux • Windows

Field	Description
Define the Rule (Python)	Use Python to define the custom detection logic. This section supports syntax highlighting and validation (IntelliSense) to help users create accurate and efficient rules. Use the default code in this box as a reference or starting point. **IMPORTANT:** The custom Python scripts are intended to be executed exclusively for compliance checks and validations. To ensure the scripts are used properly and no security risks or unintended changes occur, the system implements the following restrictions and safeguards: • Only a predefined set of Python libraries and functions required for compliance checks are available for use. Libraries or functions that enable writing, deleting, or creating operations are excluded. • Only authorized users with specific permissions can create or update custom scripts. This ensures that only trusted individuals can define compliance checks.

6. For Compliance Violation Severity, define the severity level of the compliance violation to ensure proper categorization and prioritization. Possible values are:

   - Critical
   - High
   - Medium
   - Low
   - Informational

7. For Compliance Controls, assign the rule to one or more existing compliance controls.

   NOTE:

   Only Custom Detection Rules (not built-in rules) can be assigned to custom controls.

   a. Click Add.

   b. Select a custom compliance control from the list.

   c. Click Assign.

8. For Remediation, you can optionally define the remediation steps to address any detected misconfiguration.

9. Click Create.

   The new rule appears in the Rules List.

   You can now use the rule as a check to either create an issue or monitor adherence to a specific requirement.

   Create an issue

   Under Posture Management → Policies → Cloud Workload, add the Custom Detection Rule to a Policy. This policy automatically runs the rule and creates an issue if the check fails.

   Monitor compliance adherence

   Under Posture Management → Compliance → Catalogs → Standards, create a custom standard that includes the custom control associated with the Custom Detection Rule, and then create an assessment profile that runs the custom standard. You can then monitor the compliance results in a report. For more information, see Monitor and track compliance adherence.

8.3.5 | Use an existing rule to create a new Custom Detection Rule

1. Navigate to **Posture Management** → **Rules & Policies** → **Rules** → **Cloud Workload**.
2. In the **Cloud Workload Rules** page, click the policy you want to enable or disable.
3. In the **Details** page, click the **More Options** icon (⋮) and then select **Save as new**.
4. Modify the fields as required.
5. Click **Create** to create the new custom detection rule.

8.3.6 | Edit a Custom Detection Rule

1. Navigate to **Posture Management** → **Rules & Policies** → **Rules** → **Cloud Workload**.
2. In the **Cloud Workload Rules** page, click the rule you want to edit.
3. In the **Details** page, click the **More Options** icon (⋮) and then click **Edit**.
4. Make the necessary changes.
5. Click **Update** to save the changes.

8.3.7 | Delete a Custom Detection Rule

1. Navigate to Posture Management → Rules & Policies → Rules → Cloud Workload.
2. In the Cloud Workload Rules page, click the rule you want to delete.
3. In the Details page, click the More Options icon (⋮) and then select Delete.
4. In the confirmation message, click Delete.

9 | Create prevent policy for Kubernetes clusters

Learn how to create and apply custom cloud workload prevent policies for Kubernetes clusters using Cortex Cloud.

> **PREREQUISITE:**
>
> **Ensure that the Kubernetes Connector is installed for this Kubernetes cluster and that admission controller enforcement is enabled.**

1. Create a dynamic asset group for a Kubernetes cluster

An asset group is a collection of assets based on shared attributes. A dynamic asset group uses filters to group current and future assets that meet the defined criteria. Create a dynamic asset group that targets a specific Kubernetes cluster in Cortex Cloud.

1. Select Inventory → Assets → Groups → Add Group.
2. Define a meaningful Group Name that represents the group's purpose to improve usability.
3. Select the Kubernetes Resource Cluster filter and the operator Contains. Enter the name of your target Kubernetes cluster.
4. Click Create Dynamic Group to save and apply the asset group configuration.

2. Create a custom rule for the Kubernetes Connector

A cloud workload rule defines the criteria for identifying security violations. Custom rules allow you to define and implement tailored security and compliance checks within cloud workloads. These rules enable organizations to detect specific conditions, vulnerabilities, or misconfigurations that might not be covered by built-in system rules. Create a custom workload rule that is applied specifically to Kubernetes environment variables scanned by the Kubernetes Connector.

1. Select Posture Management → Rules & Policies → Rules → Cloud Workload → Create Custom Rule.
2. Define a meaningful Rule Name that represents the rule's purpose to improve usability. Optionally add a description of the rule to explain its purpose.
3. Under Scanner, select Kubernetes Connector.
4. Under Kubernetes Resources, select All to apply the rule across all supported resources.

5. Under Define The Rule (Rego), use Rego to define the custom detection logic. You can use the default code in this box as a reference or starting point. More information on how to use Rego syntax is available at the Open Policy Agent site.

6. Under Compliance Violation Severity, define the severity level of the compliance violation based on the impact or criticality of the rule.

7. Click Create.

   The new rule appears in the Rules List. You can now use the rule to create a custom compliance policy.

3. Create a custom cloud workload compliance policy

Cloud workload policies help you prevent and manage security violations in your cloud runtime instances. They enable you to apply detection logic to specific asset groups at the desired SDLC stage, and define what action needs to be taken if the conditions are met. Create a misconfiguration policy that includes the custom rule you created for the Kubernetes Connector and is applied to the asset group you created.

1. Select Posture Management → Rules & Policies → Policies → Cloud Workload.

2. In the Cloud Workload Policy page, click + Create Policy and select Misconfiguration policy type.

3. In the General page, define a meaningful Policy Name. Optionally add a description outlining the purpose of the policy. Click Next.

4. In the Conditions page, use the filter to find and select the custom rule you created. Click Next.

5. In the Scope page, select the asset groups that correspond to the Kubernetes clusters where the policy should be enforced. Click Next.

6. In the Action page, under Select an Action, select Prevent and Create an Issue.

7. Under Issue Severity, define the severity level according to your policy enforcement requirements.

8. Click Done to finalize and apply the new cloud workload compliance policy.

4. Verify Kubernetes admission controller policy enforcement

After the custom cloud workload policy has been applied, verify that it has been successfully pushed to your Kubernetes cluster and is functioning as expected.

1. Verify the cloud workload rule was deployed in Cortex Cloud

   Log into an instance that has access to your Kubernetes cluster and list the panrules deployed in the pan namespace:

   ```
   kubectl get panrules -n pan
   ```

Output example:

```
NAME                                         AGE
e18734e2-a3b2-5e2d-a255-d73b302caee01        5m
```

The rule name in the output should match the ID of the rule created above.

> **NOTE:**
>
> If there are no rules listed, check that the cloud workload rule and policy were correctly configured and applied in Cortex Cloud.

2. Verify asset scope mapping

   Ensure that the asset scope is mapped to the intended Kubernetes cluster by checking the rule contents:

```
kubectl get panrules 2d1fd464-9d94-4ba8-aa6d-6fcf576d0045 -n pan -o yaml | grep -i assetScope
```

   Output example:

```
assetScope:
eyJBTkQiOlt7IlNFQVJDSF9GSUVMRCI6InhkbS5rdWJlcm5ldGVzLnJlc291cmNlLmNsdXN0ZXIiLCJTRUFSQ0hfVFlQR
SI6IkNPTlRBSU5TIiwiU0VBUkNIX1ZBTFVFIjoiZmFhaG1lZC1jb3J0ZXgtY2xvdWQtZGVtbyJ9XX0
```

   Decode the asset scope, which is base64-encoded:

```
echo
eyJBTkQiOlt7IlNFQVJDSF9GSUVMRCI6InhkbS5rdWJlcm5ldGVzLnJlc291cmNlLmNsdXN0ZXIiLCJTRUFSQ0hfVFlQR
SI6IkNPTlRBSU5TIiwiU0VBUkNIX1ZBTFVFIjoiZmFhaG1lZC1jb3J0ZXgtY2xvdWQtZGVtbyJ9XX0= | base64 -d
```

   Output example:

```
{
  "AND": [
    {
      "SEARCH_FIELD": "xdm.kubernetes.resource.cluster",
      "SEARCH_TYPE": "CONTAINS",
      "SEARCH_VALUE": "jsmith-cortex-poc"
    }
  ]
}
```

   Ensure that the "SEARCH_VALUE" matches the name of the cluster you are targeting.

3. Test policy enforcement

   Deploy a sample that violates the custom policy (for example, using an image without the `:latest` tag). This is a sample `nginx.yml` file:

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx-deployment
  labels:
    app: nginx
spec:
  replicas: 3
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
      - name: nginx
        image: nginx:1.20
        ports:
        - containerPort: 80
```

Apply the deployment:

```
kubectl apply -f nginx.yml
```

Expected error:

```
Error from server: error when creating "nginx.yml": admission webhook "admission-control-
validating-config.pan.svc" denied the request: image 'nginx:1.20' can’t deployed old image in
Namespace default.
```

This confirms that the admission controller is working correctly and is enforcing the custom policy.

10 | What's new in Kubernetes Connector?

This topic describes the changes, additions, known issues, and fixes for each version of the Kubernetes Connector. If Auto Upgrade is enabled in your Kubernetes Connector, you will automatically enjoy the latest released features without having to manually upgrade to the new version.

Kubernetes Connector releases

Cortex Cloud supports the following current Kubernetes Connector versions. Click the link to view the new features, addressed issues, and known issues per release.

Release Version	Release Notes	Release Date
1.3	Kubernetes Connector version 1.3	Nov 9, 2025
1.2	Kubernetes Connector version 1.2	July 20, 2025

Kubernetes Connector version 1.3

New features

The following section describes the new features introduced in Kubernetes Connector version 1.3.

Feature	Description
Unified Kubernetes Onboarding	Streamlined Kubernetes onboarding process in a single, easy-to-use wizard. Now you can discover all available security capabilities based on your license, configure everything in one flow, and deploy your entire solution with one consolidated installer.
Kubernetes Connector	Supports AKS, EKS, GKE, managed OpenShift, self-managed Kubernetes vanilla clusters, and self-managed OpenShift with a Kubernetes Native installation method of Helm Installer. For more details see Supported Kubernetes distributions.
KSPM Dashboard	A visual overview of your Kubernetes security posture. It includes inventory insights, protection coverage, most vulnerable clusters, malware and secrets detected, and more.
Compliance standards	Enjoy out-of-the-box CIS compliance standards for Kubernetes environments (CIS EKS, CIS GKE, CIS AKS, CIS OpenShift, and CIS Kubernetes).
Secret, malware, and vulnerabilities	Generate secret, malware, and vulnerabilities posture issues by declaring policies on Kubernetes clusters

Known limitations

The following table describes known limitations in the Kubernetes Connector release.

Feature	Description
Connector onboarding and cluster identifier	The Kubernetes Connector automatically calculates the Kubernetes cluster cloud identifier by using the metadata service (for EKS and GKE) and cluster resources (for AKS). • For EKS and GKE, the metadata service must be enabled.

Kubernetes Connector version 1.2

New features

The following section describes the new features introduced in Kubernetes Connector version 1.2.

Feature	Description
Kubernetes Connector Onboarding	Supports AKS, EKS, GKE, managed OpenShift, and self-managed Kuberntes Vanilla clusters, with a Kubernetes Native installation method of Helm Installer.
KSPM Dashboard	A visual overview of your Kubernetes security posture. It includes inventory insights, protection coverage, riskiest clusters, and more.
Compliance standards	Enjoy out-of-the-box CIS compliance standards for Kubernetes environments (CIS EKS, CIS GKE, CIS AKS, CIS OpenShift, and CIS Kubernetes).

Feature	Description
Secret, malware, and vulnerabilities	Generate secret, malware, and vulnerabilities posture issues by declaring policies on Kubernetes clusters
Kubernetes internet exposure	r

Known limitations

The following table describes known limitations in the Kubernetes Connector release.

Feature	Description
Connector onboarding and cluster identifier	The Kubernetes Connector automatically calculates the Kubernetes cluster cloud identifier by using the metadata service (for EKS and GKE) and cluster resources (for AKS). • For EKS and GKE, the metadata service must be enabled.

11 | Supported Kubernetes distributions

The following are the supported Kubernetes platform versions for the Kubernetes connector (Posture Management). The table shows the latest version that is supported. We support n-3 versions of each supported Kubernetes environment.

Kubernetes Environment	Notes
Managed clusters	• Amazon Elastic Kubernetes Service (EKS) • Microsoft Azure Kubernetes Service (AKS) • Google Kubernetes Engine (GKE)
Managed OpenShift	Managed Openshift clusters, including ROSA (Red Hat OpenShift on AWS), are supported.
Self-Managed	We support every CNCF-certified Kubernetes solution. We've tested our solution on: • Self-managed vanilla/on-premise Kubernetes clusters. • Self-managed OpenShift Kubernetes clusters.

The following are the Kubernetes platforms that are supported with Cortex XDR agents (Real-time protection).

This table shows the Kubernetes platform versions that have been compatibility tested. The table shows the latest version that has been tested. All versions that are not EOL, up to the latest version are supported.

Linux Kubernetes Platform	Version
Unmanaged Kubernetes (k8s)	1.30
Amazon Elastic Kubernetes Service (EKS)	1.33

Linux Kubernetes Platform	Version
BottleRocket OS x86_64 User mode agent only	
BottleRocket OS aarch64 User mode agent only	
Microsoft Azure Kubernetes Service (AKS)	1.33
CBL-mariner 2 x86_64	
Google Kubernetes Engine (GKE)	1.33
Google Container-Optimized OS (COS)* x86_64 User mode agent only	
Google Kubernetes Engine (GKE) Autopilot	
Oracle Kubernetes Engine (OKE)	1.33
Red Hat Openshift Container Platform (OCP)	4.16

Linux Kubernetes Platform	Version
RHCOS* x86_64 User mode agent only	
SUSE Rancher Kubernetes Engine 2 (RKE2)	1.28
Talos	1.8.3

NOTE:

In Google Container-Optimized OS release 100 and earlier, where the FANOTIFY EXEC flag is not supported, the Kernel configuration may be partial for the user mode agent to properly function. In such cases, the agent will fallback to asynchronous mode.

In RHCOS version 4.12 and earlier, the Kernel configuration may be partial for the user mode agent to properly function. In such cases, the agent will fallback to asynchronous mode.