Palo Alto Prisma Cloud Compute and Red Hat OpenShift

Table of Contents

Overview

In this lab, you will use Prisma Cloud Compute to secure runtime aspects of a Red Hat® OpenShift® Cluster.

This lab will cover:

- An overview of Prisma Cloud Compute features and architecture
- Accessing the Prisma Cloud Compute Console
- Going through a real world use case using a demo application and Prisma Cloud Compute

Prerequisites

In order to participate in the lab, you will need access to the Red Hat Product Demo System (RHPDS) Prisma Cloud Compute for OpenShift cluster OpenShift CLI client ("oc") installed on your local machine.

Prisma Cloud Compute - at a glance

Prisma™ Cloud Compute (PCC) Edition delivers a cloud workload protection platform (CWPP) for modern enterprises, providing holistic protection across hosts, containers, and serverless deployments in any cloud, throughout the software lifecycle. Prisma Cloud Compute Edition is cloud native and API-enabled, protecting all your workloads regardless of their underlying compute technology or the cloud in which they run.

Features

Rounding out its holistic protection, Prisma Cloud Compute Edition offers:

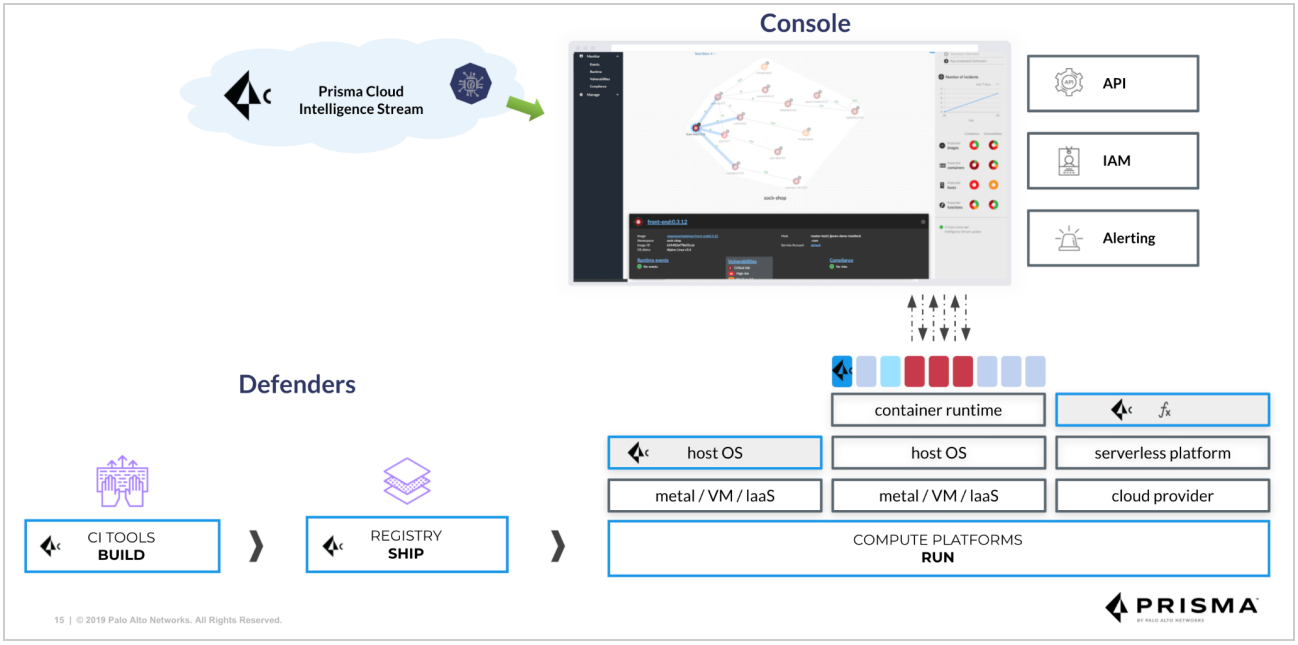
- **Vulnerability management.** Enjoy security from development through production with unmatched vulnerability detection, understanding, and prevention at every stage of the application lifecycle.
- **Compliance.** Easily implement and maintain compliance for Docker, Kubernetes, and Linux CIS Benchmarks as well as external compliance regimes and custom requirements, including the industry's first compliance checks for the Istio® service mesh.
- **CI/CD security.** Integrate security directly into the continuous integration (CI) process to find and fix problems before they ever make it into production.
- **Runtime defense.** Secure your environments at scale with machine learning that automatically creates least-privileged, allow-list-based runtime models for every version of every application.
- **Web application and API security.** Protect against Layer 7 and OWASP Top 10 threats in any public or private cloud.
- **Access control.** Establish and monitor access control measures for cloud workloads and cloud native applications across underlying hosts, Docker, and Kubernetes while integrating with identity and access management (IAM) and secrets management tools, along with other core technologies.

How it works

Prisma Cloud Compute Edition provides flexible deployment options to protect your workloads and applications wherever you choose to deploy them. Defenders, agents deployed within your environments, protect standalone VMs, Docker containers, Kubernetes clusters, CaaS, PaaS apps on Red Hat® Openshift® Container Platform, and serverless applications. Defenders protect by allow-listing application behavior and preventing anomalous actions from occurring. Defense in depth combines core cloud native firewalling with runtime defense to protect east-west traffic flows and leverage machine learning for known application behavior.

Prisma Cloud Compute Edition provides vulnerability management and compliance for the full software lifecycle by integrating with any CI process, Docker registry, code repository, or production environment to continuously monitor risk with powerful risk factors and prioritization. Enterprise-grade access control capabilities govern all cloud resources across compute infrastructure, secrets, Kubernetes audits, and IAM tooling.

Prisma Cloud Compute Architecture

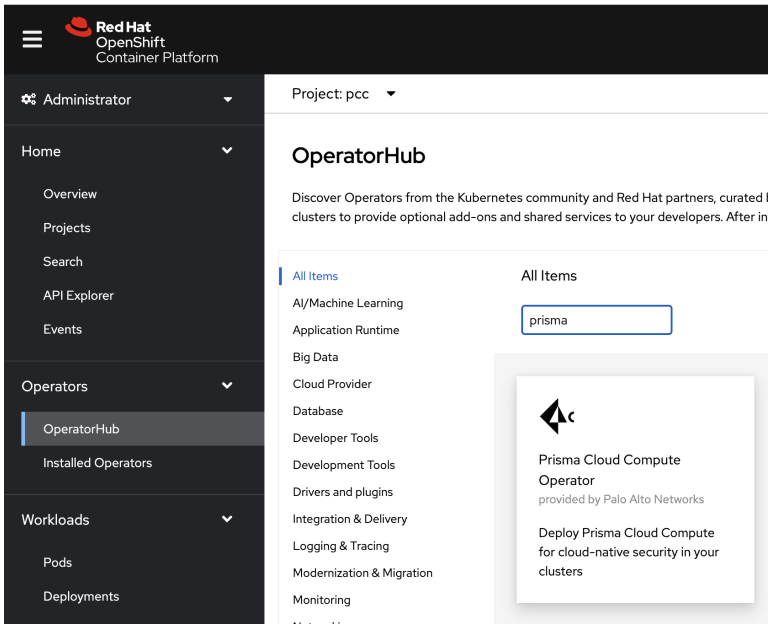


Module 1: Deploy Prisma Cloud Compute via Operator

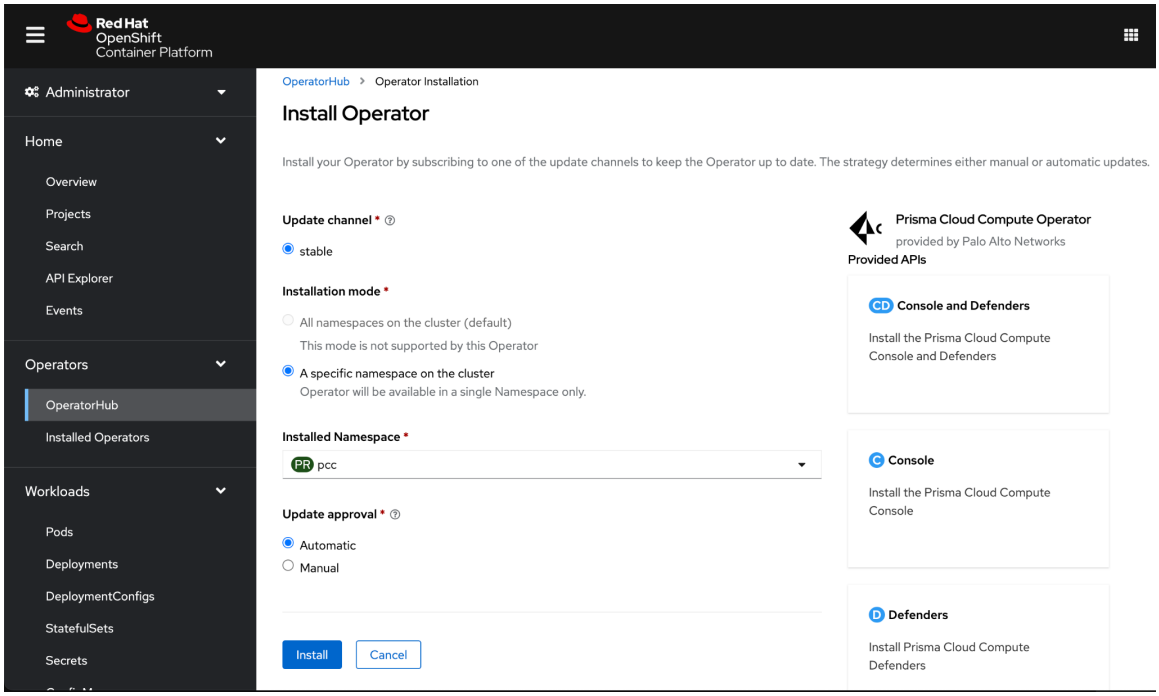
You will be provided a link and login credentials from your lab instructor to the OpenShift and Prisma Cloud Compute consoles. Let's get started.

1. Create a project named "pcc" (Prisma Cloud Compute).

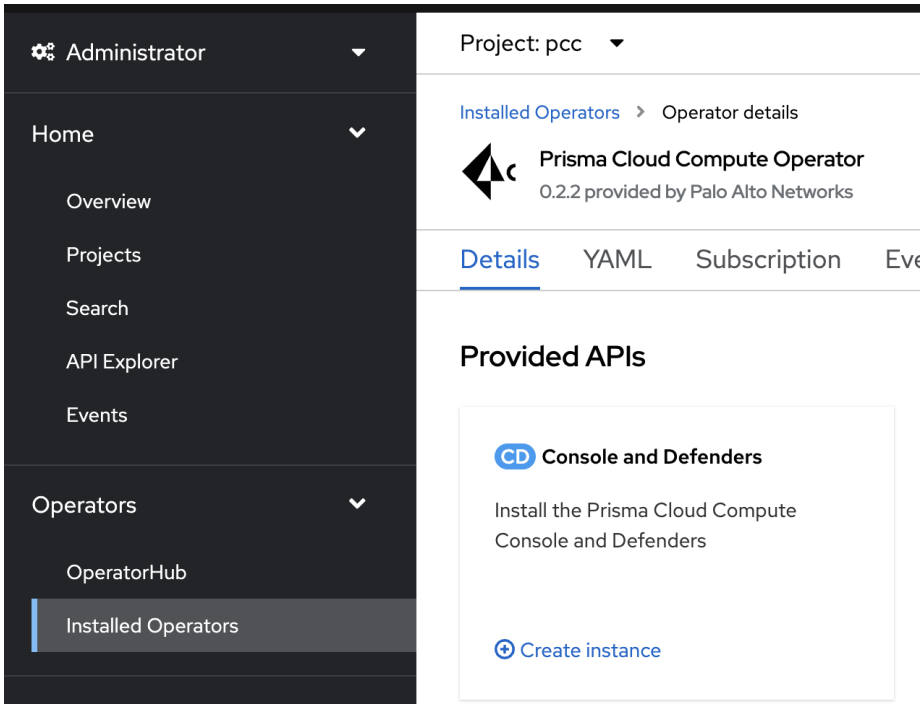
2. Navigate to **Operators > OperatorHub** and search for "prisma". Select the certified operator (not *Community*). In the subsequent pop-up window, click **Install**.



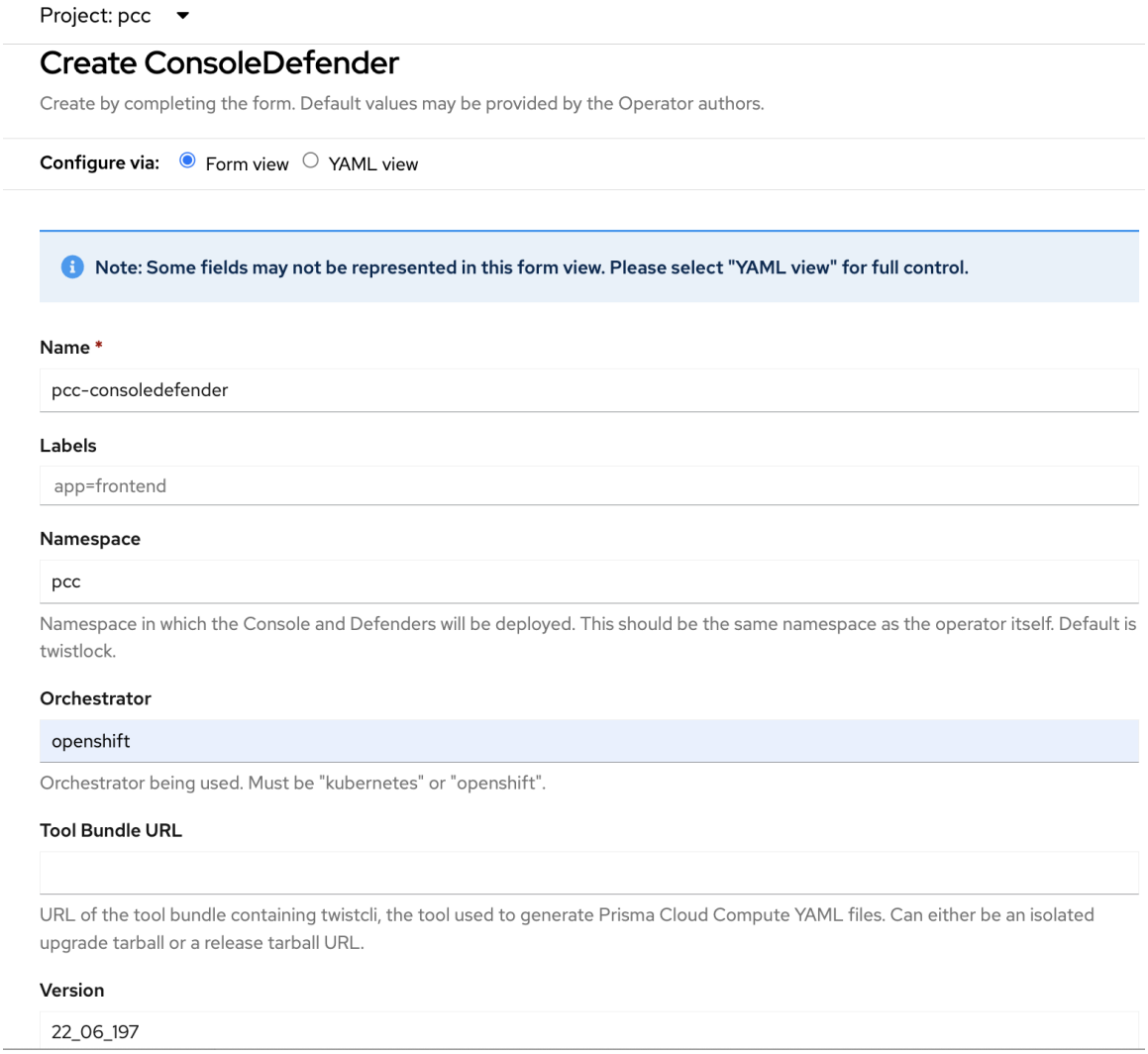
3. Install the operator in the "pcc" namespace.



4. In the OpenShift console, navigate to **Operators > Installed Operators**. Under **Details > Provided APIs**, select *Create Instance* from the "Console and Defenders" tile.



5. In the **Create ConsoleDefender** wizard, enter the following information into the provided fields:
    - Namespace = pcc
    - Orchestrator = openshift
    - Version = 22_06_197

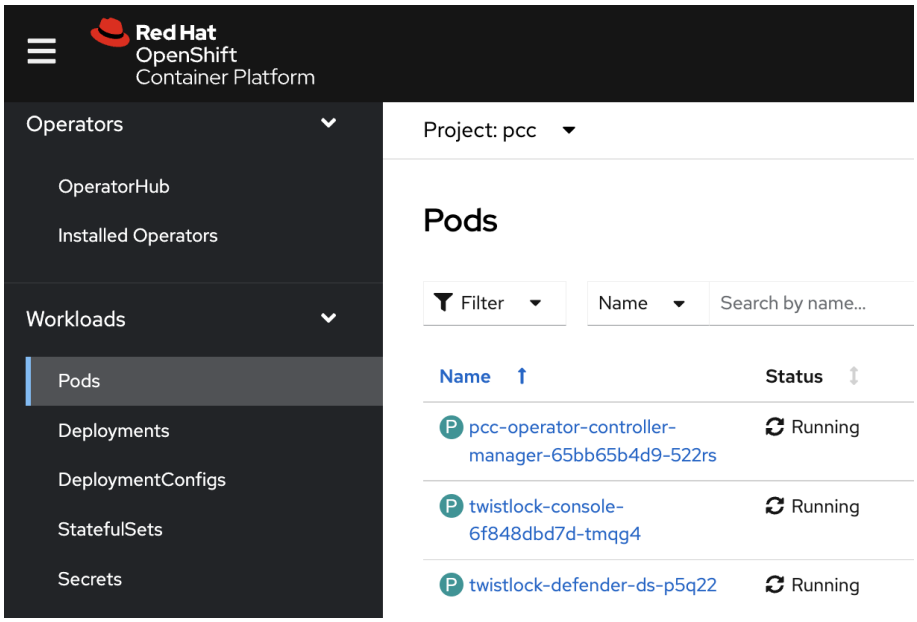


6. For the **Credentials** portion of the form, *Access Token* and *License* will be provided in the workshop (RH employees, send an email request to techbd@paloaltonetworks.com if outside of the workshop).

7. Enter a *Username* and *Password* of your choosing - this will be used to access the console. (Note: it is recommended to change the password after first login)

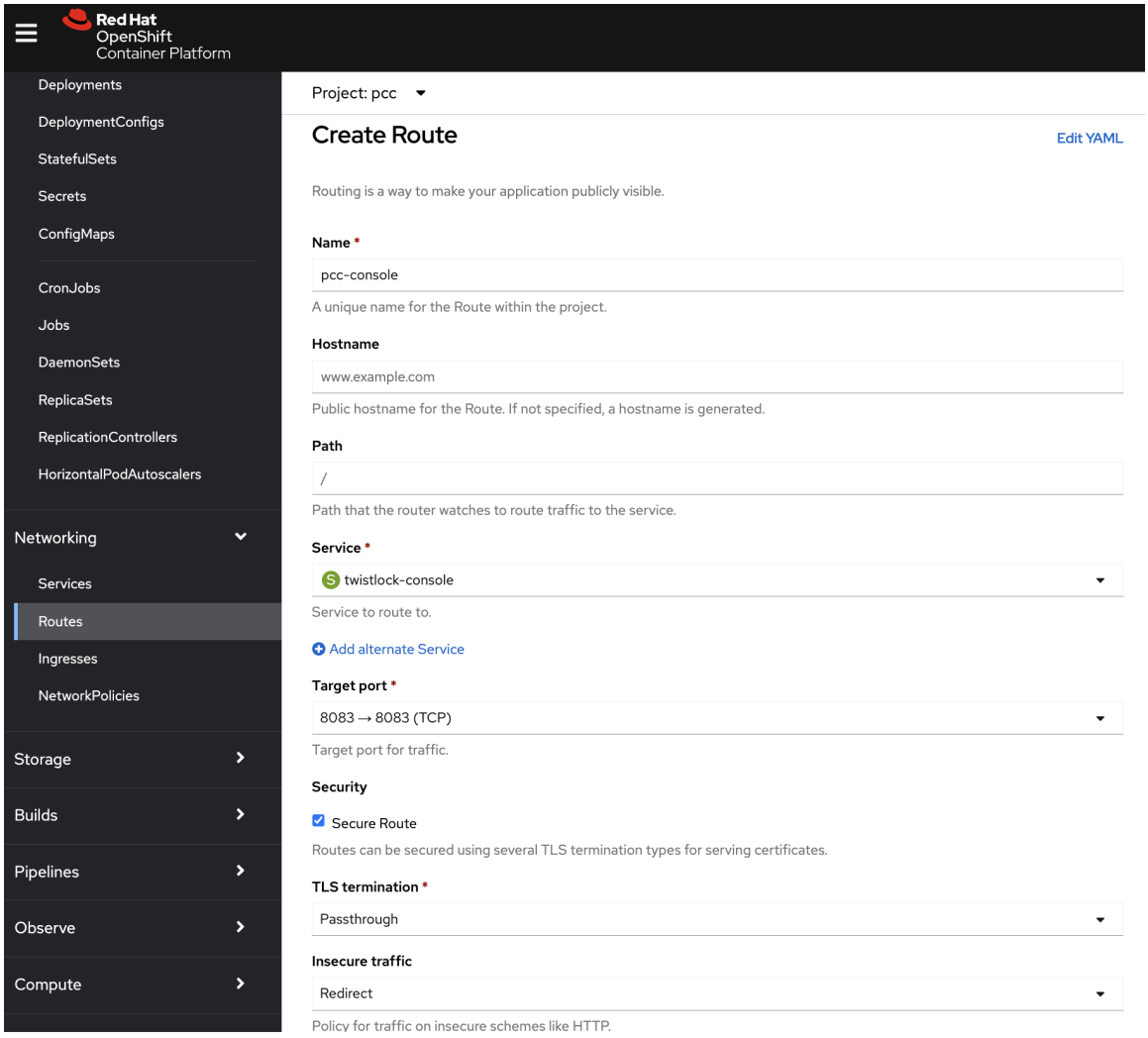
8. Leave the **Console Installation Options** and **Defender Installation Options** blank to assume default configuration. Click **Create**.

9. Navigate to **Workloads > Pods** to check that the operator deployed resources.

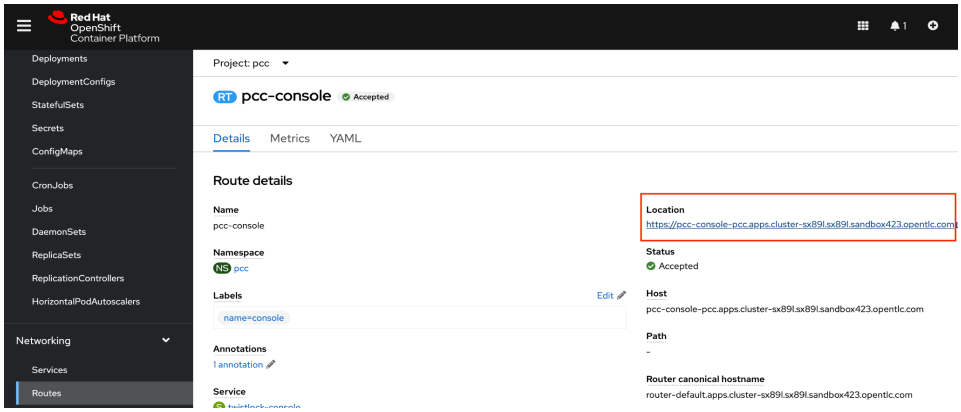


10. To access the console, we need to create a route within OpenShift. Navigate to **Networking > Routes** and enter the following into provided fields:

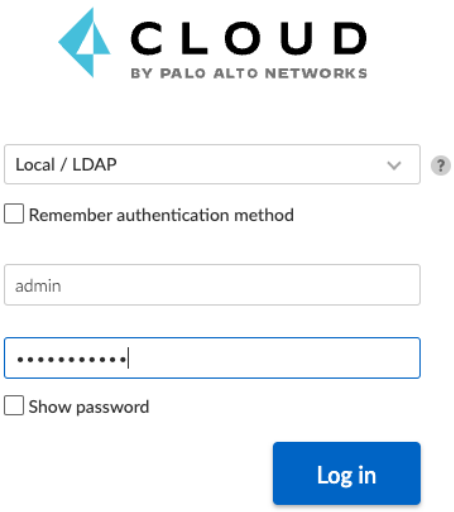
    - Name = pcc-console
    - Service = twistlock-console (from dropdown)
    - Target Port = 8083 (from dropdown)
    - Enable Secure Route
    - TLS Termination = Passthrough
    - Insecure Traffic = Redirect



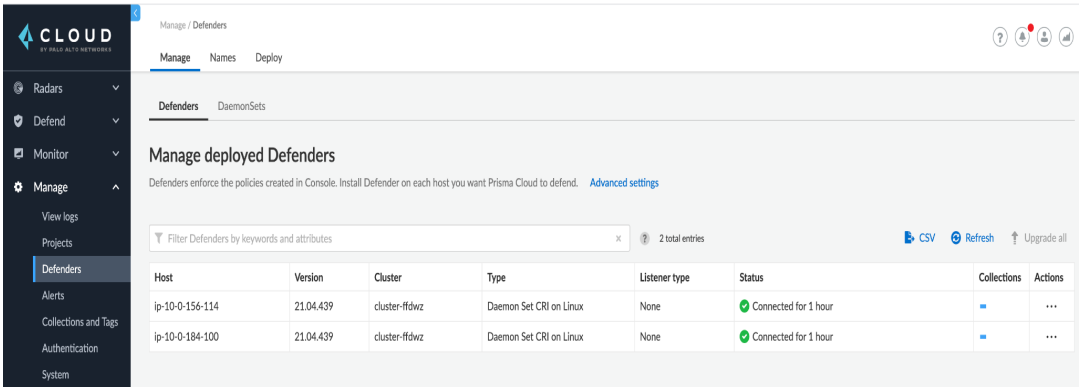
11. Within **Networking > Routes,** click the *pcc-console* route. Click the console URL under **Location** to login to Prisma Cloud Compute.



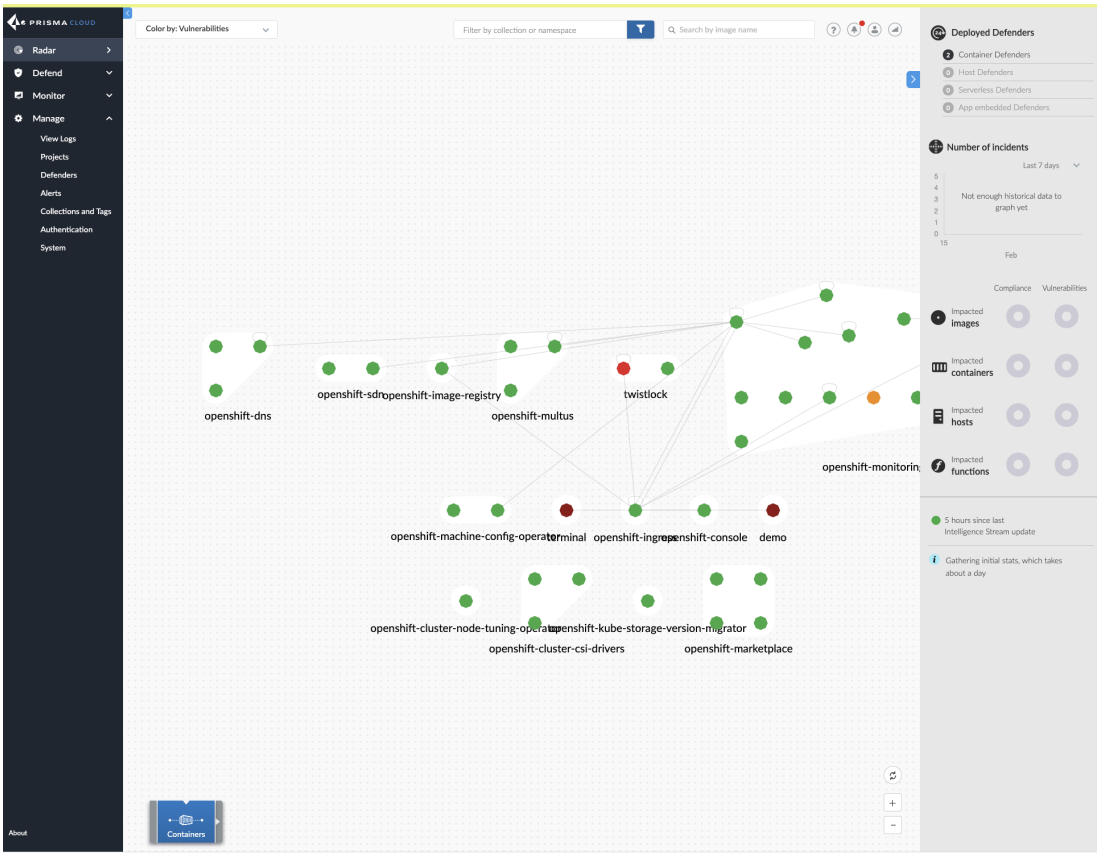
12. Enter the username and password chosen in step 6.



13. In the Prisma Cloud Console, navigate to **Manage** > **Defenders** > **Manage** to see a list of deployed defenders. You will see something similar to the following:



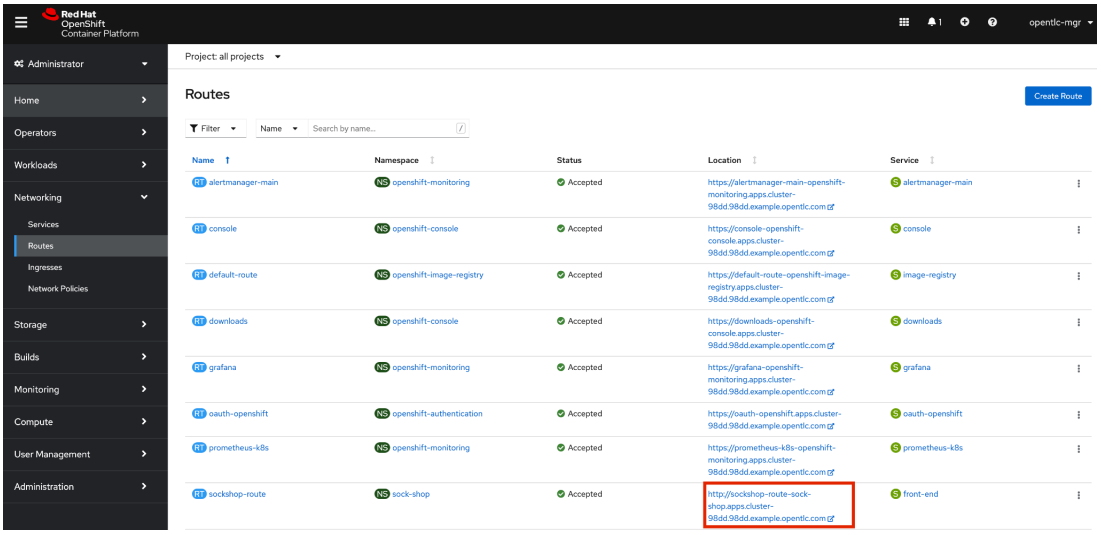
14. Navigate to the **Radar** View on the left menu and select **Container** from the bottom menu option. You will see the defender has begun to scan the existing environment and populate the Console with information.



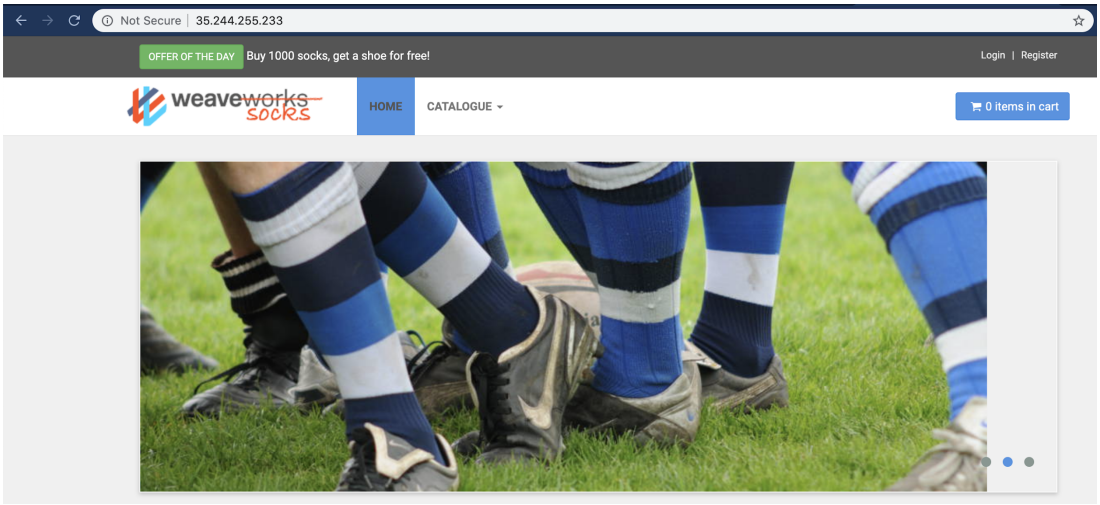
Module 2: Sock Shop - Microservices Demo Application

Now, we're ready to deploy some applications on our cluster. We will use Sock Shop, which emulates a typical service we would see on the internet. For this lab we have installed Sock Shop for your use.

15. Verify the **sockshop-route** location is available via the OpenShift console, navigate to **Networking** > **Routes** to see a list of routes available, you will see something similar to the following:

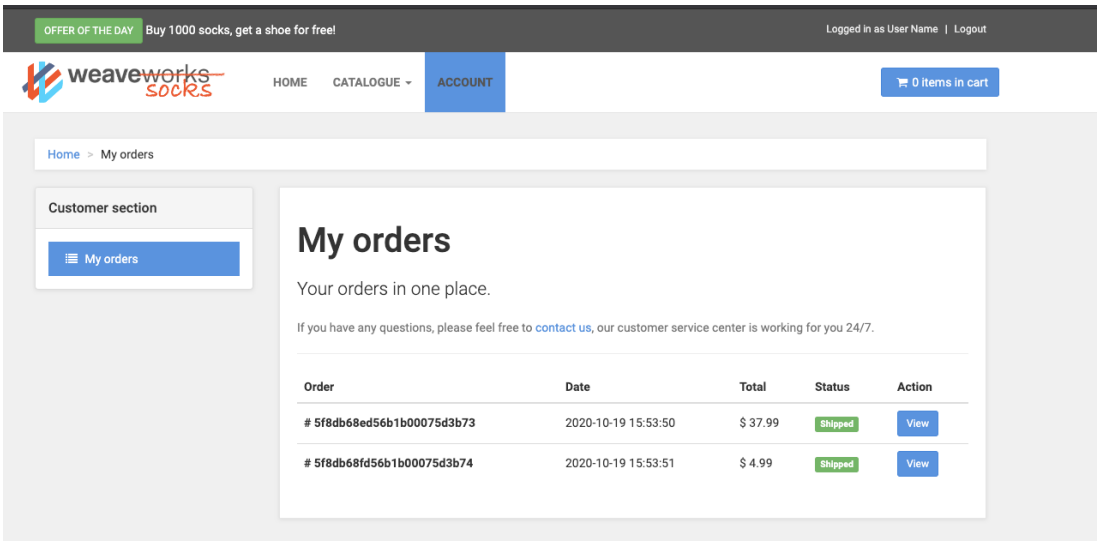


16. Click on the **sockshop-route** Location URL. A new window will pop-up and you should see the following:



17. Register on the SockShop website with the fake credentials (*All information including email, names, address, credit card info should be fake*):
    Username: **user**
    Password : **password**

18. Browse around the site and purchase some items. Create a fake credit card account (*use at least 5 digits for the card number*), and shipping address. By doing this you are generating communication traffic. This will allow Prisma Cloud Compute to learn communication paths between the containers.

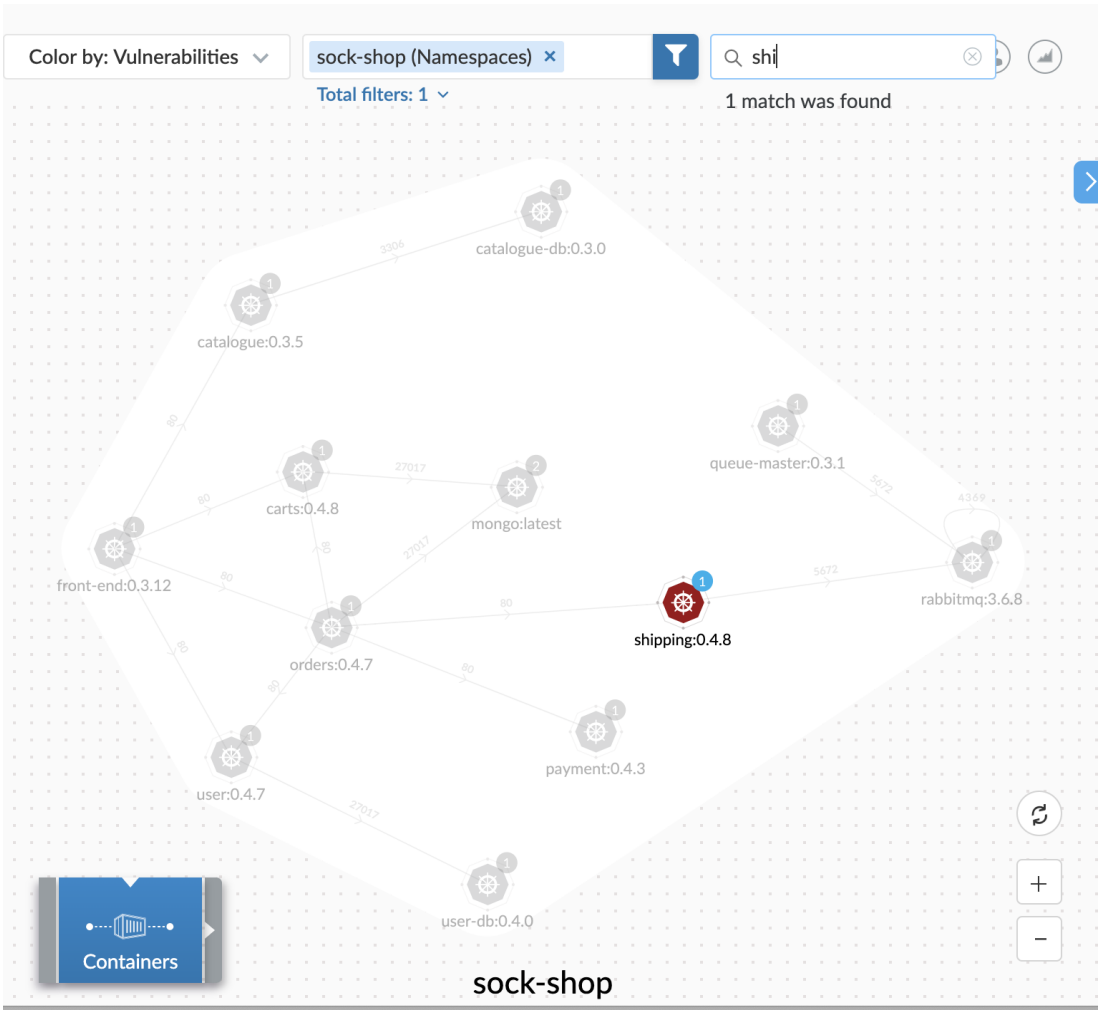


19. Go to **Radars** > **Setting** > **Container network monitoring**, and toggle the switch to enable.

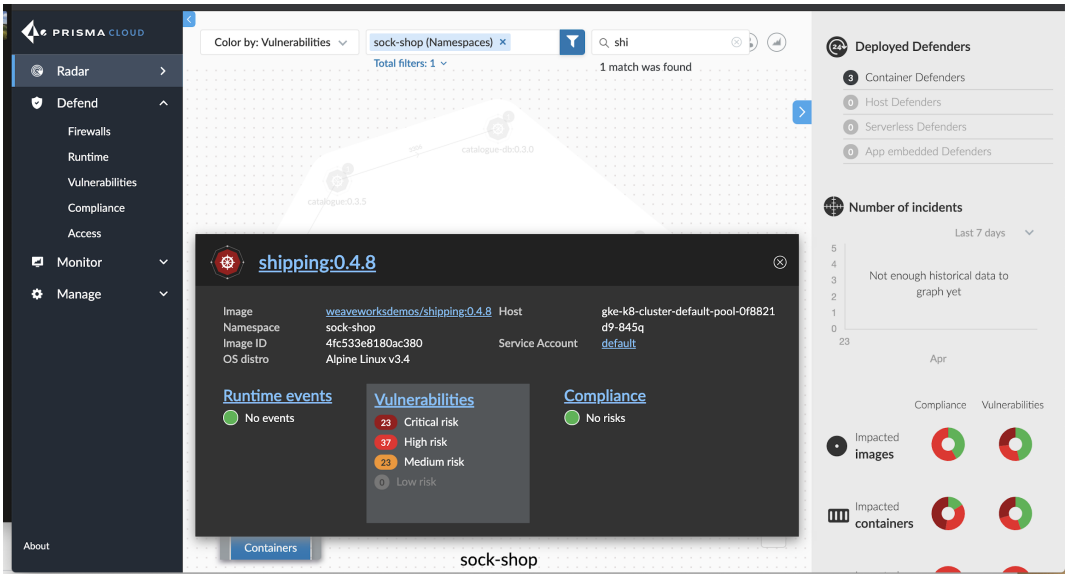
20. Click on **Radars** > **Containers**, and check sock-shop in the filter, then refresh in the lower right-hand corner. Additional container services will be visible as well as the communication paths.

21. In the Prisma Cloud Compute console, view all the services that are running in the deployments created so far: Click the **+ or -** sign at lower right corner to adjust the view. This represents the Sock Shop website vulnerable containers.

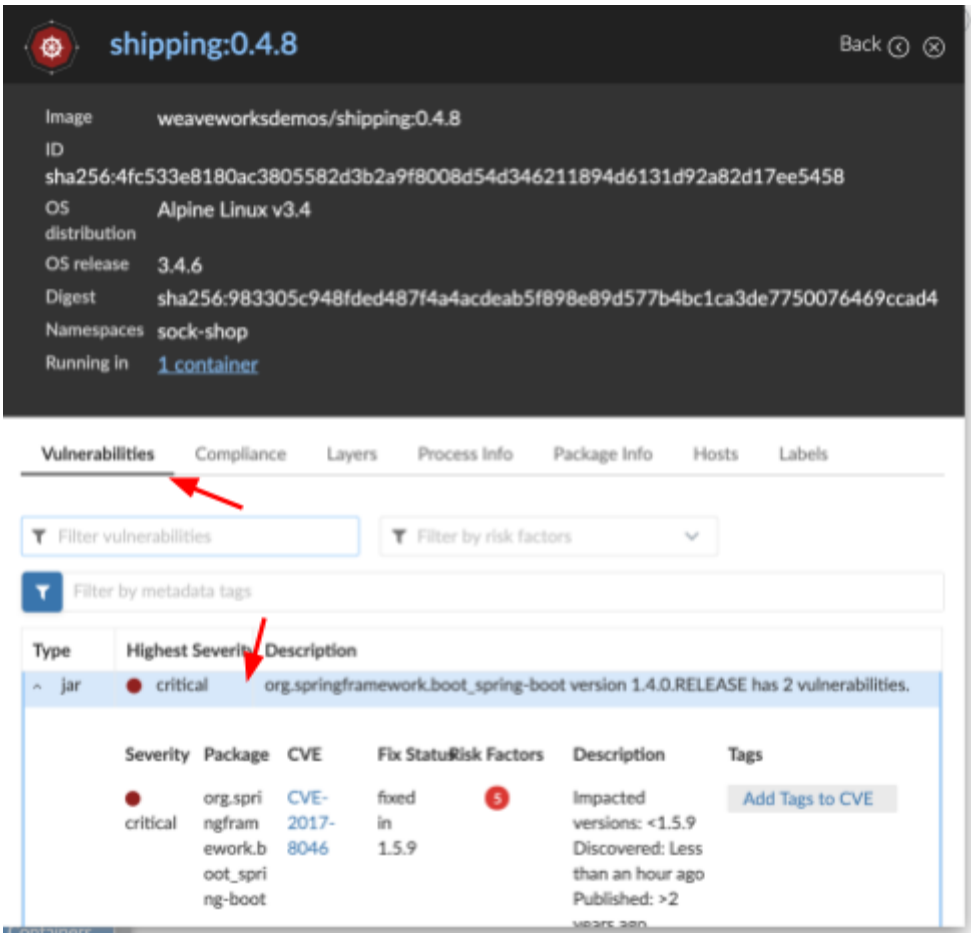
22. Type **"ship"** in the search box to search for a container name starting with **"ship"**.



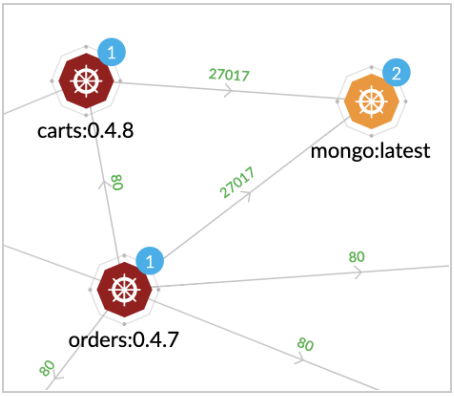
23. Click the container named **shipping:0.4.8** to see all the information and alerts Prisma Cloud Compute has found that are associated with this container:



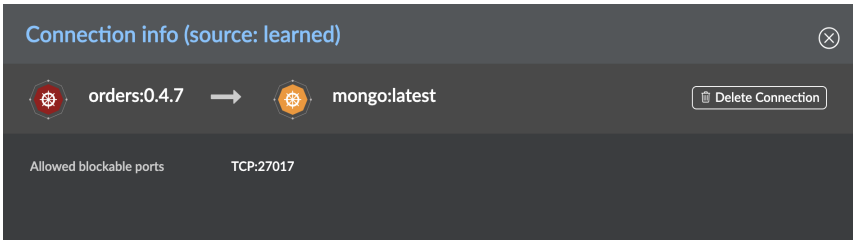
24. Click **Vulnerabilities**. Click on one of the lists to expand it and list its associated CVE:



25. Close the container details, clear the search, and go back to Radar view. Now find the container **mongo:latest** and observe the network connections to the container from container **order:0.4.7**



26. Click on the link between **order:0.4.7** and **mongo:latest** and notice:
    - The traffic is using TCP port 27017
    - The direction is from **order:0.4.7** to **mongo:latest**



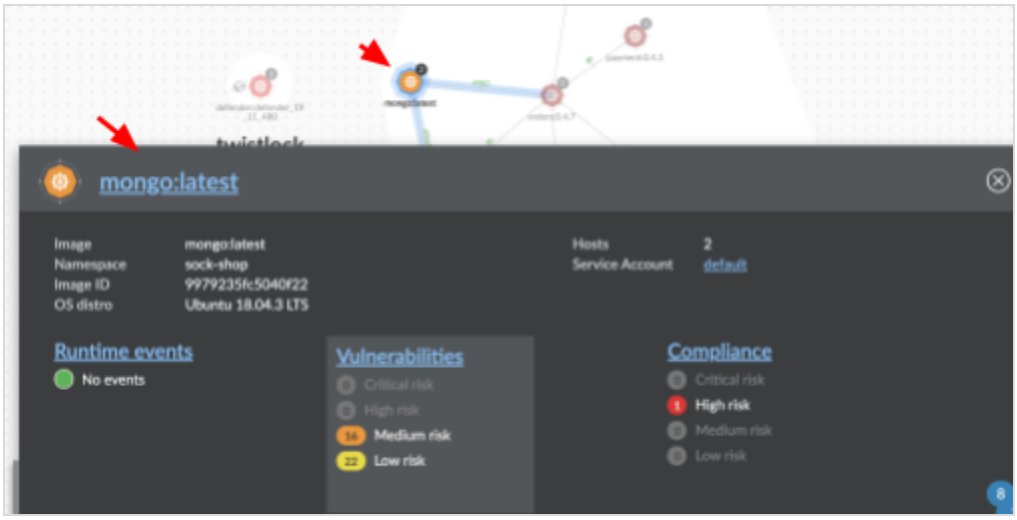
Module 3: Runtime Security and Container Runtime Model

Modern threats require layered runtime protection. When a new CVE (refer to LINK for the details on CVEs) is announced that affects a container image in your runtime environment, you need automated runtime protection that secures your entire environment. Prisma Cloud Compute is able to automatically learn and build a 4-dimension Container Model about your environment: network, file system, processes, and system calls.

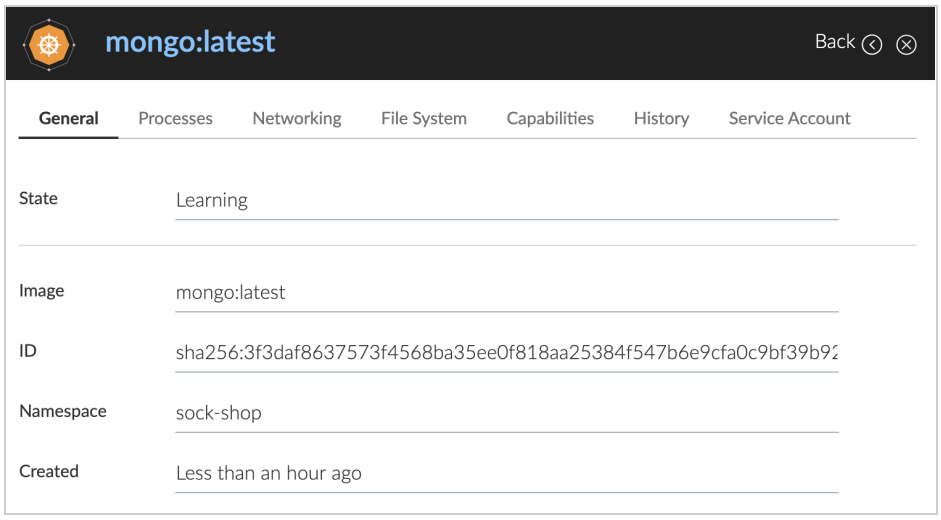
Container Models are the results of the autonomous learning that Prisma Cloud Compute performs every time it sees a new image in an environment. A model is the allow list for what a given image should be doing, across all runtime sensors. Models are automatically created and maintained by Prisma Cloud Compute. They provide an easy way for administrators to view and understand what Prisma Cloud Compute has learned about their images. Critically, models are built from both static analysis (such as building a hashed process map based on parsing an init script in a Dockerfile ENTRYPOINT) and dynamic behavioral analysis (such as observing actual process activity during early runtime of the container).

Next, let's review the Container Model of a service in Sock Shop and what Prisma Cloud Compute has learned so far:

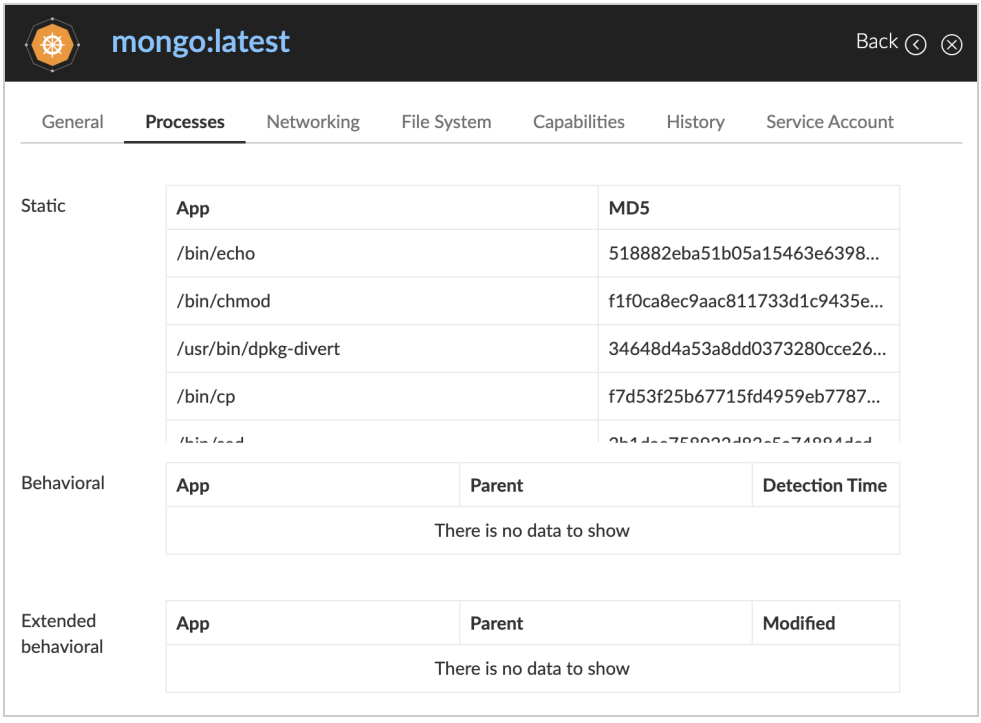
27. Select **Radar** tab in the left panel, then click on **mongo:latest** at Radar Map.
28. Click on **mongo:latest** hyperlink in the service details pop up to explore the details about the service **mongo:latest.**



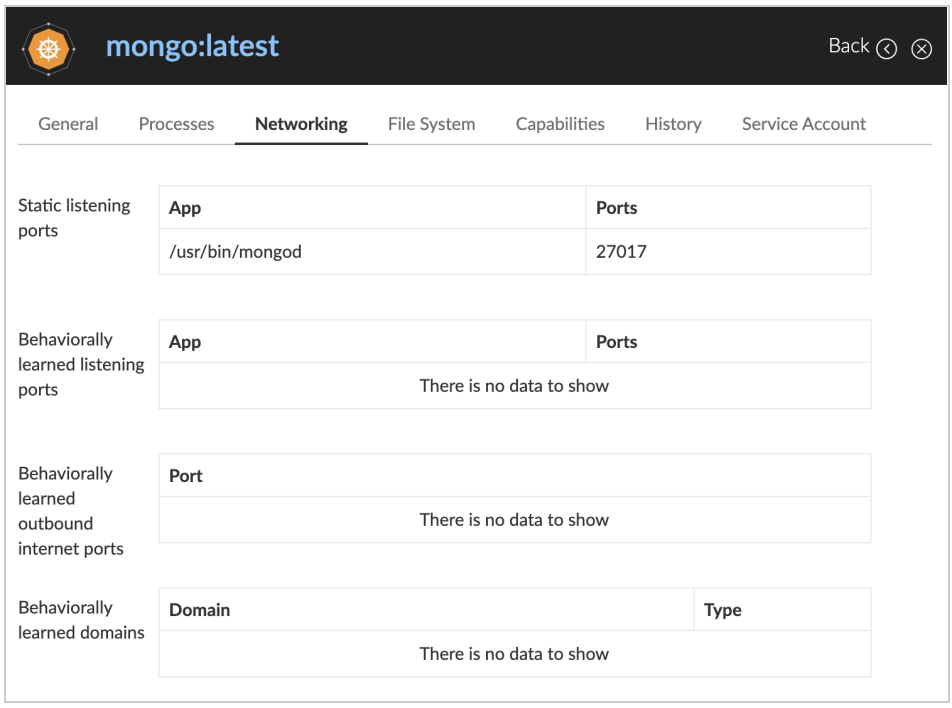
29. Review the information under the **General** tab.



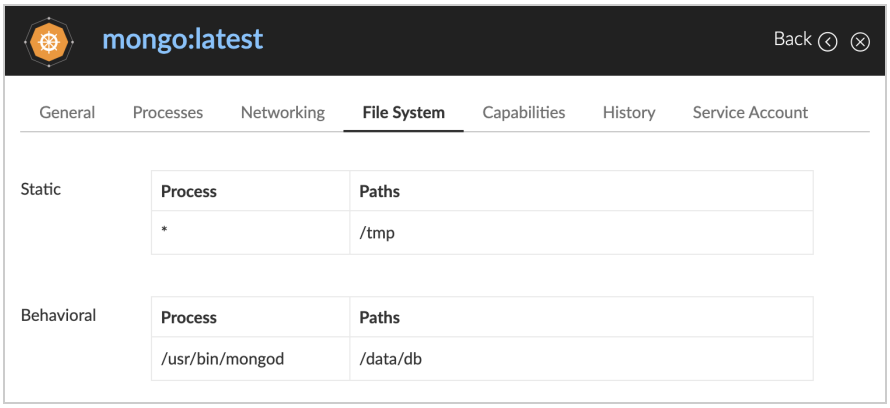
30. Click the **Process** tab to review all processes run by **mongo:latest**.



31. Click on the **Networking** tab to review the learned networking ports, internet connections, and domains that are used by **mongo:latest**. You can see there is no internet connection, and the internal listening port is 27017.



32. Click on the **File System** tab to review learned file system mount points accessed by **mongo:latest**.



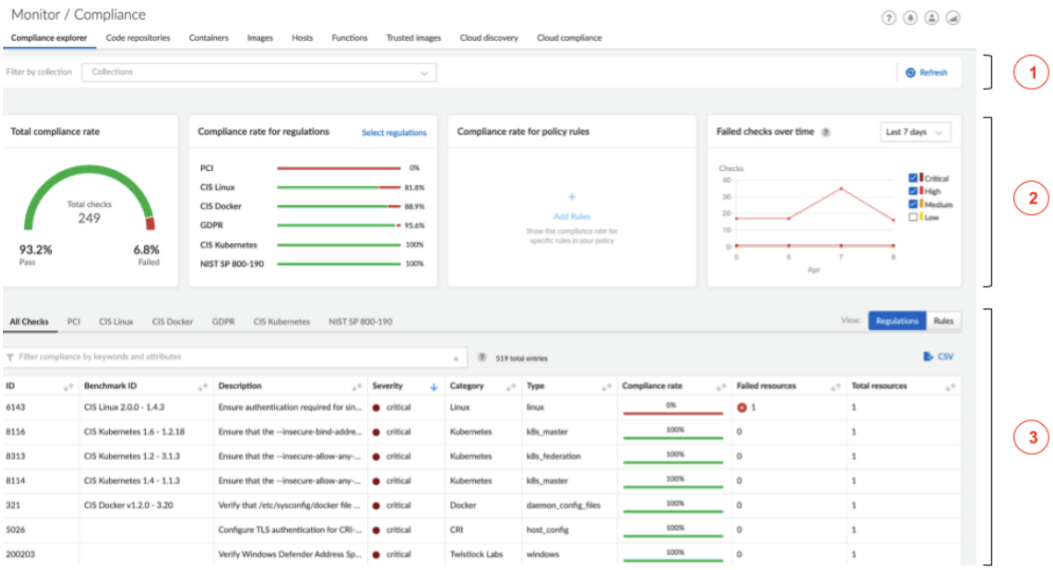
You have reviewed the container runtime model learned by Prisma Cloud Compute. After the learning period, any process, network activity, file system access, or system call beyond the Container Model can be detected as anomalous behavior. You have the option to set a policy to alert, prevent, or block the anomalous behavior.

Module 4: Compliance & Malware Scanning

Compliance Explorer is a reporting tool for compliance rate. Metrics present the compliance rate for resources in your environment on a per-check, per-rule, and per-regulation basis. Report data can be exported to CSV files for further investigation.

The key pivot for Compliance Explorer is failed compliance checks. Compliance Explorer tracks each failed check, and the resources impacted by each failed check. From there, you can further slice and dice the data by secondary facets, such as collection, benchmark, and issue severity.

Go to **Monitor > Compliance > Compliance Explorer**



1. **Collection filter.** Collections group related resources together. Use them to filter the data in Compliance Explorer. For example, you might want to see how all the entities in the sock-shop namespace in your production cluster comply with the checks in the PCI DSS template.

2. **Roll-up charts.** Configurable charts that summarize compliance data for the perspectives you care about. They report the following data:

   - Total compliance rate for your entire estate.
   - Compliance rate by regulation. Click Select regulations to configure which benchmarks and templates are shown on the page. Benchmarks are

industry-standard checklists, such as the CIS Docker Benchmark. Templates are Prisma Cloud-curated checklists Checks are selected from the universe of checks provided in the product that pertain to directives in a regulatory regime, such as the Payment Card Industry Data Security Standard (PCI DSS).

- **Compliance rate by rule.** Provides another mechanism to surface specific segments of your environment when scrutinizing compliance. Click Add rule to configure the card.
- **Historical trend chart for compliance rate.** Shows how the compliance rate has changed over time.

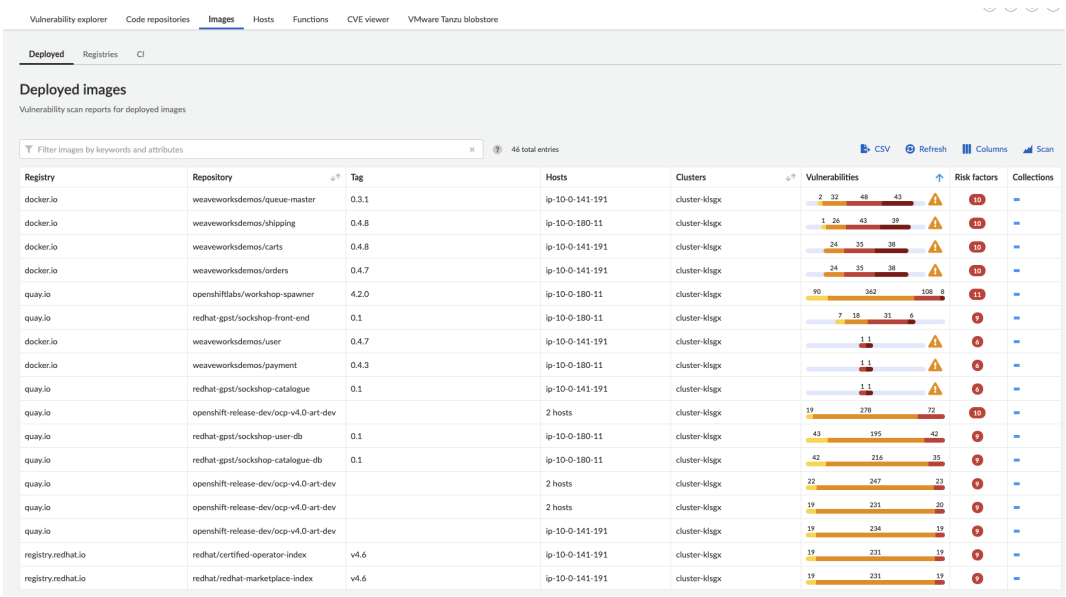
The lists in the regulation and rule cards are sorted by compliance rate (the lowest compliance rate first).

3. **Results table.** Shows the universe of compliance checks, and the compliance rate for each check, relative to:
   - Your policies and the checks that are enabled. Every compliance check has an ID, and it's either enabled or disabled.
   - The collection selected at the top of the page.
   - The filters applied (e.g. show critical severity issues only.)

Besides detecting software vulnerabilities (CVEs) and compliance issues (such as images configured to run as root), Prisma Cloud also detects malware in your container images. No special configuration is required to enable this feature.

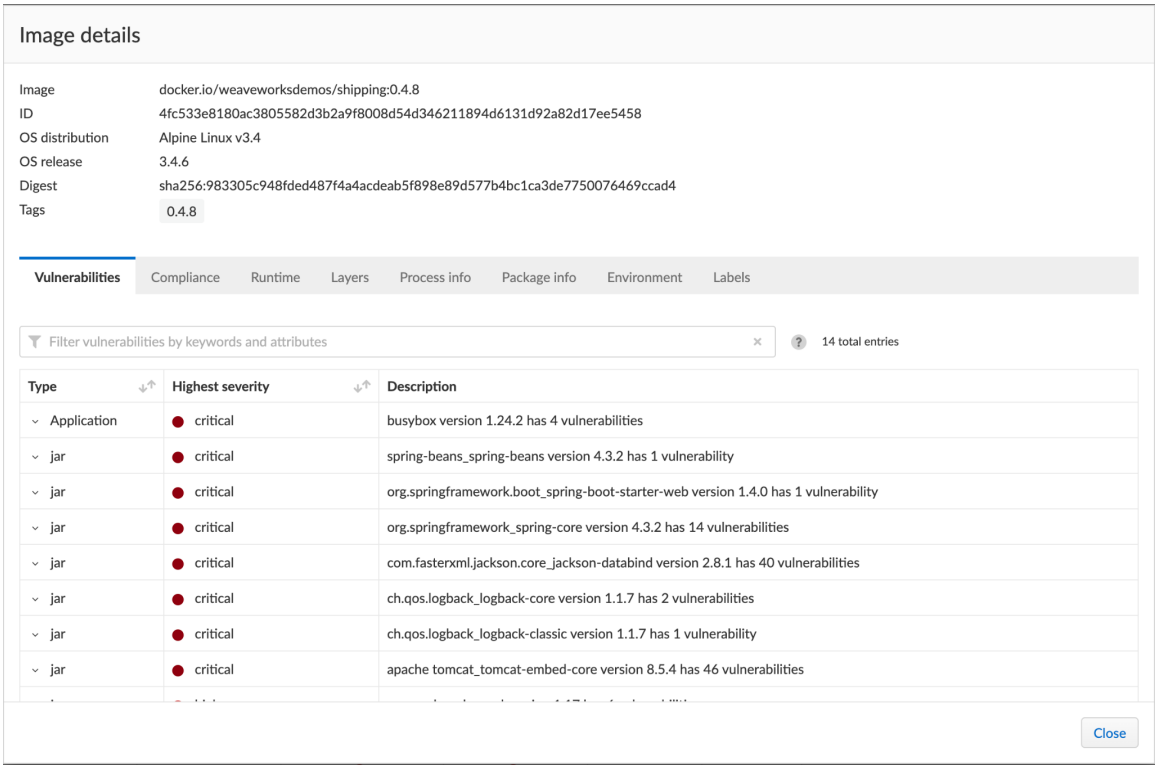
Malware data is sourced from commercial providers, Prisma Cloud Labs, and open source lists. The image scanner looks for malware in binaries in the image layers, including the base layer.

Go to **Monitor > Vulnerabilities > Images**



Click on an image to get a detailed report from the last image scan.

In the detailed report, click on the **Vulnerabilities** tab.



NOTE: You may not have same results as screenshot.

Module 5: Runtime Defense - ATT&CK Explorer

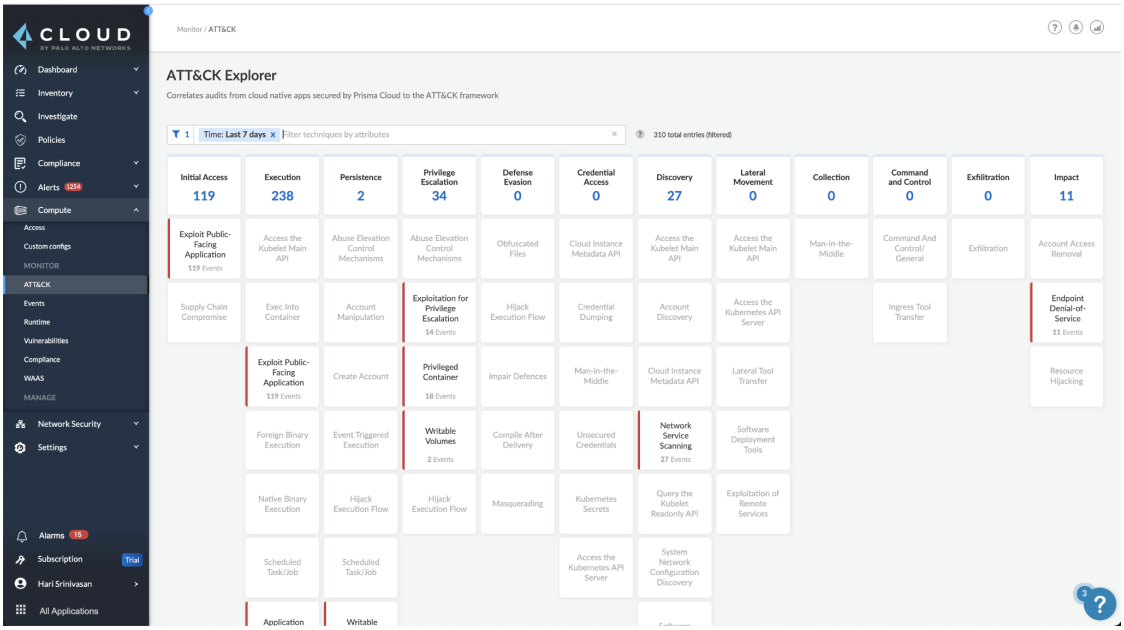
Prisma Cloud's monitoring section includes an Att&CK Explorer dashboard, providing a framework that helps you to contextualize runtime audits, manage them, and generate risk reports.

ATT&CK Explorer is a knowledge base of tactics and techniques that adversaries use to attack applications and infrastructure. It's a useful framework for threat-informed defense, where a deep understanding of adversary tradecraft can help protect against attacks.

The ATT&CK framework has two key concepts:

- **Tactics:** An adversary's technical goals.
- **Techniques:** How those goals are achieved or What they achieve

Go to **Monitor > ATT&CK**



The ATT&CK dashboard serves as a portal to the raw events in the **Monitor > Events** view. All Prisma Cloud audits are mapped to the tactics and techniques in the ATT&CK framework. For example, when Defender detects a crypto miner in your environment, we map the audit to the `Resource Hijacking` technique under the `Impact` tactic.

The ATT&CK dashboard collates audits, maps them to the tactics and techniques, and presents the data visually in the ATT&CK matrix. Each card in the matrix shows a count of events. Higher counts represent higher severity issues. Filters let you slice and dice the data to inspect specific segments of your environment.

You may not show any count for this lab.

Module 6: Alerts

Prisma Cloud lets you surface critical policy breaches by sending alerts to any number of channels. Alerts ensure that significant events are put in front of the right audience at the right time. You can create any number of alert profiles, where each profile gives you granular control over which audience should receive which notifications.

Go to **Manage > Alerts**

- Alert mechanism
- AWS Security Hub
- Cortex XSOAR
- Email
- Google Cloud Pub/Sub
- Google Cloud SCC
- IBM Cloud Security Advisor
- JIRA
- PagerDuty
- ServiceNow Security Incident Response
- ServiceNow Vulnerability Response
- Slack
- Webhook

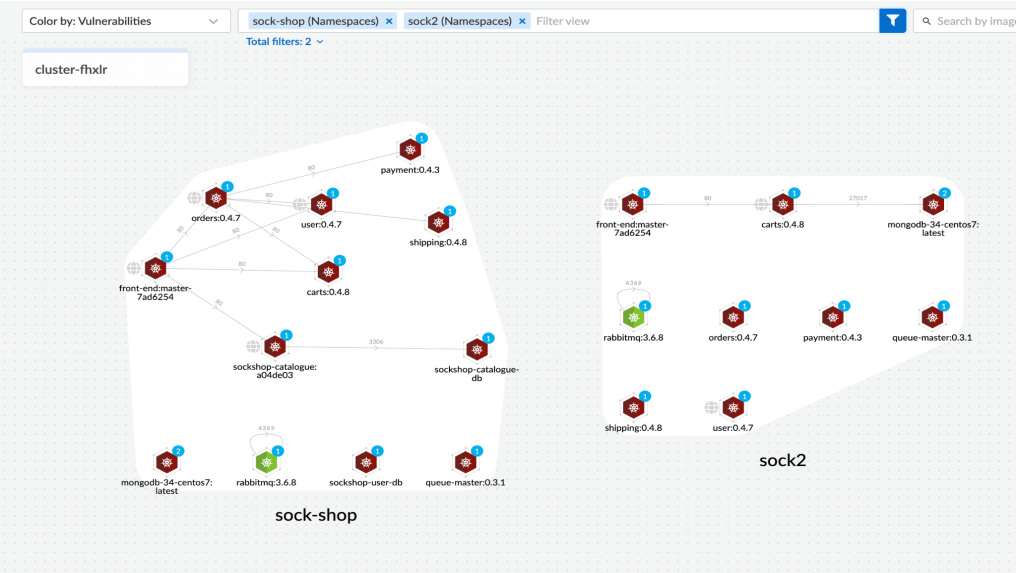
For this lab, we will not enable Alerts.

Module 7: Block Container Images

In this portion of the workshop we will prevent a container image from running on a pod using Vulnerability Rules and/or Admission Control (Open Policy Agent).

Objective: block container images that contain the string "sockshop" within the image name, (AND have Critical CVE findings)

Outcome: a second instance of the sock shop application is deployed with 3 less pods than the built-in example.



Option 1) Block Images with Vulnerability Rules:

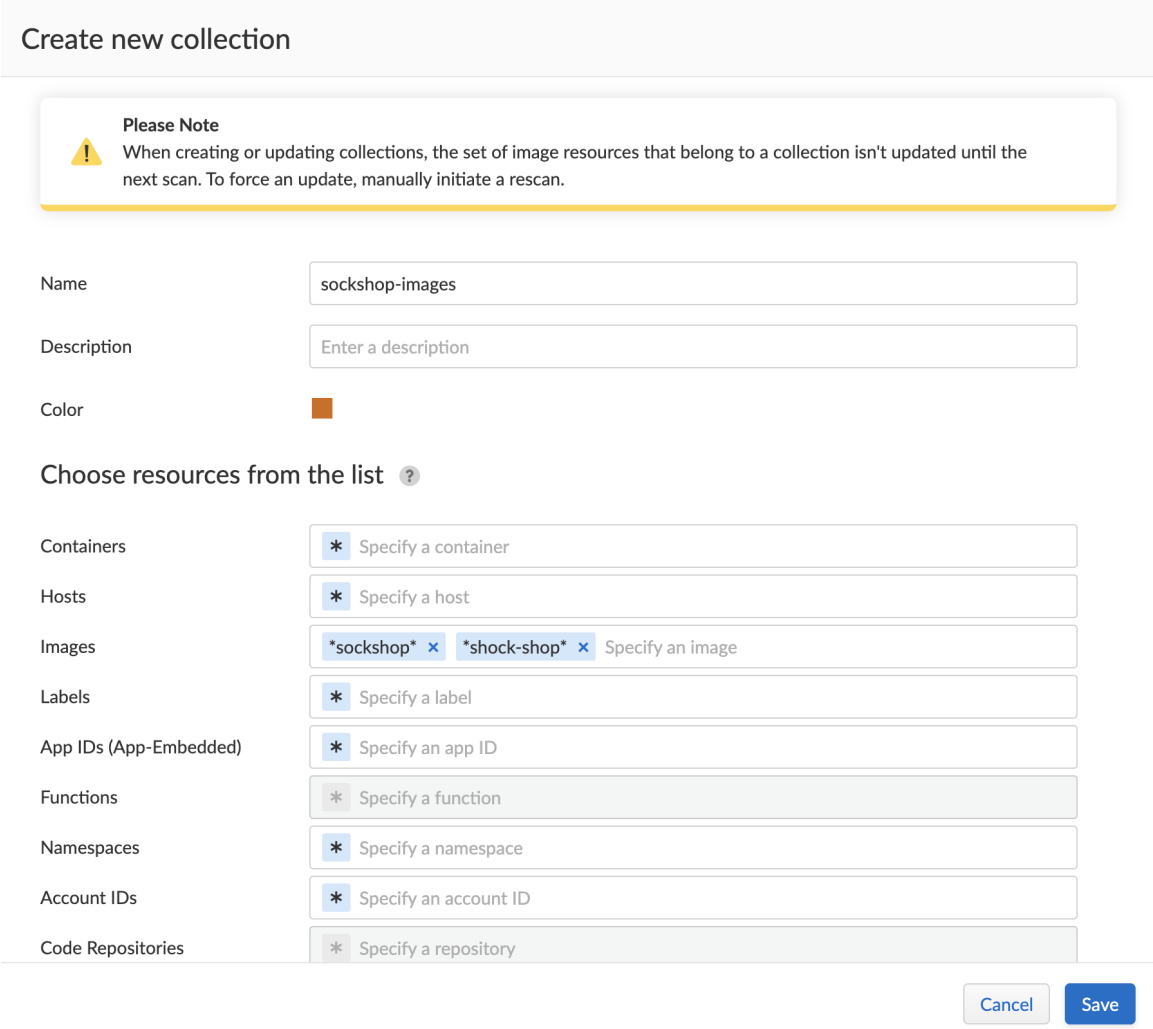
Vulnerability policies are composed of discrete rules. Rules declare the actions to take when vulnerabilities are found in the resources in your environment. They also control the data surfaced in Prisma Cloud Console, including scan reports and Radar visualizations.

Rules let you target segments of your environment and specify actions to take when vulnerabilities of a given type are found.

Setup Doc:
https://docs.paloaltonetworks.com/prisma/prisma-cloud/22-06/prisma-cloud-compute-edition-admin/vulnerability_management/vuln_management_rules

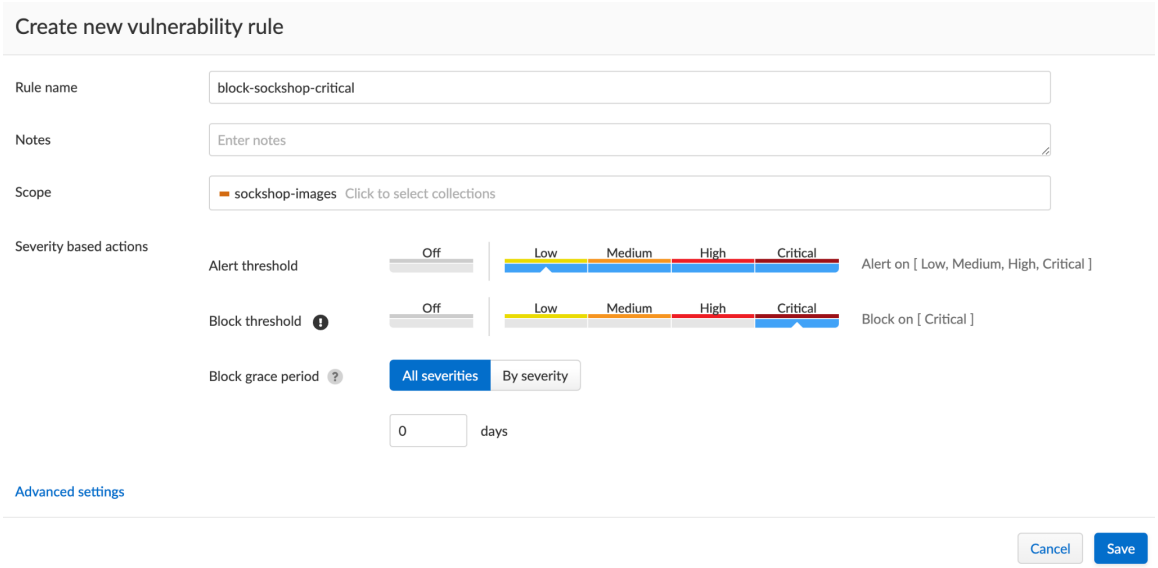
Instructions:

1. Navigate to **Defend > Vulnerabilities > Images > Deployed** and select **Add Rule**
2. Add a Rule Name (*e.g. block-sockshop*) and click **Scope**
3. Uncheck the default *All* collection and click **Add Collection**
4. Enter the following details and click **Save**



5. Click **Select Collections** to save the Rule's scope

6. Update the **Block Threshold** to *Critical* and click **Save**



7. Create a second sock-shop namespace and redeploy the application
    a. oc login
    b. oc new-project sock2
    c. git clone https://github.com/mosuke5/microservices-demo-openshift.git
    d. oc apply -f microservices-demo-openshift/complete-demo.yaml
8. Verify images were blocked
    a. oc get events (and/or check OpenShift/Prisma Console)

Option 2) Block Images with Admission Controller / Open Policy Agent:

Note: if you are testing both options, ensure to *Disable the Vulnerability Rule* created in Option 1 and remove the sockshop application you deployed.

Prisma Cloud provides a dynamic admission controller for Kubernetes that's built on the Open Policy Agent (OPA). In Console, you can manage and compose rules in Rego, which is OPA's native query language. Rules can allow or deny (alert or block) pods. Console pushes your policies to Defender, which enforces them. Decisions made by the system are logged.

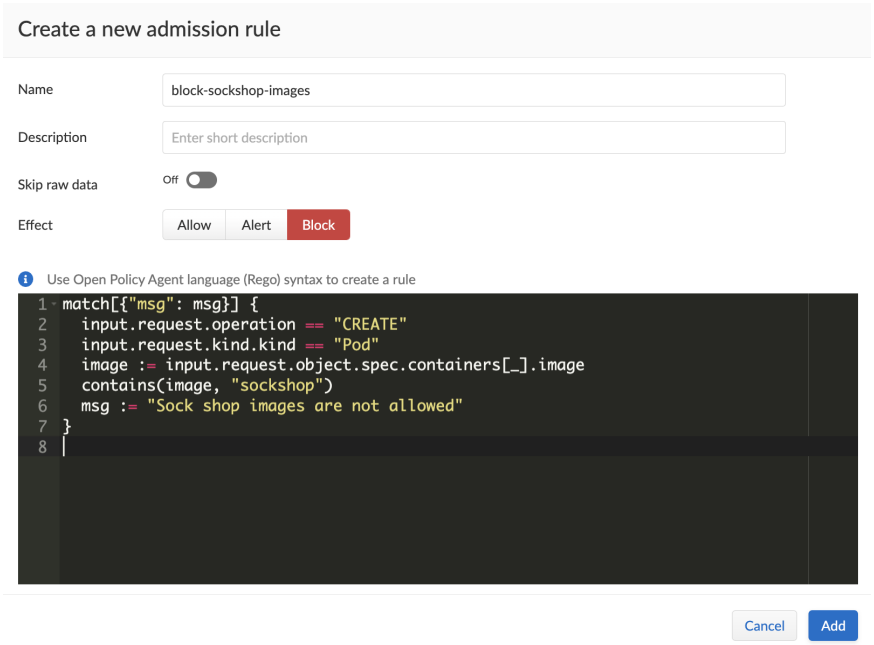
Setup Doc:
https://docs.paloaltonetworks.com/prisma/prisma-cloud/21-08/prisma-cloud-compute-edition-admin/access_control/open_policy_agent

Instructions:

1. Navigate to **Defend > Access > Admission** and enable Admission Control.
2. Click the **Go to settings** hyperlink.
3. Leave above defaults, edit the webhook configuration for your desired namespace (e.g. "twistlock -> "pcc"), copy the provided configuration, and click **Save**.
4. Create a file called **webhook.yaml** and paste in copied configuration
5. Apply the webhook
    a. oc project pcc
    b. oc apply -f webhook.yaml
6. Click **Add Rule** and enter the following policy and click **Add** when finished

    Policy:

```
match[{"msg": msg}] {
  input.request.operation == "CREATE"
  input.request.kind.kind == "Pod"
  image := input.request.object.spec.containers[_].image
  contains(image, "sockshop")
  msg := "Sock shop images are not allowed"
}
```



7. Create a second sock-shop namespace and redeploy the application
    a. oc login

   b. oc new-project sock2
   c. git clone https://github.com/mosuke5/microservices-demo-openshift.git
   d. oc apply -f microservices-demo-openshift/complete-demo.yaml
8. Verify images were blocked
   a. oc get events (and/or
   b. Check Prisma Console **Monitor > Events**, then select **Admission Audits**

End of Workshop!

Learn more

- Prisma Cloud for Red Hat OpenShift
  https://www.paloaltonetworks.com/prisma/environments/red-hat-openshift
- Prisma Cloud Compute on OpenShift (10 min demo video)
  https://www.youtube.com/watch?v=KLrwrxGSr5g
- Official Prisma Cloud marketing and technical documentation
  https://www.paloaltonetworks.com/prisma/cloud
- Prisma Cloud Compute Edition Red Hat OpenShift Operator
  https://catalog.redhat.com/software/operators/detail/5e9877e83f398525a0ceb196
- Webinar: DevSecOps for cloud-native applications with Prisma Cloud
  https://www.brighttalk.com/webinar/devsecops-for-cloud-native-applications-with-prisma-cloud/