

# Prisma Cloud Compute on RedHat OpenShift

## Introduction:

Protection requirements for cloud native applications are evolving and span virtual machines, containers and serverless workloads in public and private clouds. Security and risk management leaders must address the unique and dynamic security requirements of hybrid cloud workloads

## This workshop will address:

How Palo Alto Networks Prisma™ Cloud provides built-in coverage of Red Hat CVEs directly from the Red Hat OVAL feed, and supports open container standards like CRI-O that power the Red Hat® OpenShift® Container Platform.

## Table of Contents

Prerequisites	2
Installing Prisma Cloud Compute Console	3
Installing Defender Agents to Workloads	7
Deploy the Sock Shop webpage	12
Malware Demonstration	15
Host Models	15
Capabilities	15
List of capabilities	16
Violations	19
Enabling host runtime protection	19
Anomalous app detection	19
Log inspection	19
Networking	20

DNS	20
IP connectivity	20
Access logging	20
File integrity management (FIM)	20
Links/contacts for more information:	21

---

## Prerequisites

---

- Access to RHPDS / an OpenShift cluster.
- OpenShift CLI client ('oc') installed on local machine (from OpenShift console).
- Prisma Cloud Compute Access Token and License Key (provided).
- wget installed on local machine (for mac, 'brew install wget' from terminal).

---

# Installing Prisma Cloud Compute Console

---

- Log into cluster.  
`$ oc login <your-cluster's-API-endpoint>`
- Download Prisma Cloud Compute and prepare.  
`$ wget --no-check-certificate  
https://cdn.twistlock.com/releases/6e6c2d6a/prisma_cloud_compute_edition_20_04_163.tar.gz  
$ mkdir twistlock  
$ tar xvzf prisma_cloud_compute_20_04_163.tar.gz -C twistlock/`
- Create a Persistent Volume Claim.
  - Log into OpenShift console (link provided by instructor), from sidebar: 'Storage > Persistent Volume Claims'.
  - Change project from top drop-down.
  - Select 'Create Project' and enter 'twistlock' as the name.
  - Click Create.
  - Click 'Create Persistent Volume Claim' on the top right of screen.
  - Input the values accordingly:

Project: twistlock

## Create Persistent Volume Claim [Edit YAML](#)

**Storage Class**

Storage class for the new claim

**Persistent Volume Claim Name \***

A unique name for the storage claim within the project

**Access Mode \***

☒ Single User (RWO) ☐ Shared Access (RWX) ☐ Read Only (ROX)

Access mode is set by storage class and cannot be changed

**Size \***

Desired storage capacity

☐ Use label selectors to request storage

Use label selectors to define how storage is created

[Create](#) [Cancel](#)

- Click 'Create'.
- Generate a deployment YAML file for Console.
 

```
$ <PLATFORM>/twistcli console export openshift \
  --persistent-volume-labels "app-volume=twistlock-console" \
  --service-type "ClusterIP"
```
- Enter the Access Token.
  - o Access Token: 20cb5em3t2rliaz0qtiniptexdttxem
- Deploy the console to the cluster.
 

```
oc create -f ./twistlock_console.yaml
```

**NOTE:** You can ignore the error logs saying the project and PVC already exists.

- Create external Route to Prisma Cloud Compute Console.
  - Log into OpenShift console, from sidebar: Networking > Routes.
  - Click Create Route.
  - Input the values accordingly:

The screenshot shows the Red Hat OpenShift Container Platform console interface. On the left is a sidebar menu with categories: Config Maps, Cron Jobs, Jobs, Daemon Sets, Replica Sets, Replication Controllers, Horizontal Pod Autoscalers, Networking (expanded), Storage (expanded), Builds, and Monitoring. Under 'Networking', 'Routes' is selected. The main panel shows the configuration for a route named 'twistlock-console' in the 'twistlock' project. The configuration includes fields for Name, Hostname, Path, Service, Target Port, Security (checked), TLS Termination (Passthrough), and Insecure Traffic (Redirect). At the bottom are 'Create' and 'Cancel' buttons.

Project: twistlock

**Name \***  
twistlock-console  
A unique name for the route within the project.

**Hostname**  
www.example.com  
Public hostname for the route. If not specified, a hostname is generated.

**Path**  
/  
Path that the router watches to route traffic to the service.

**Service \***  
twistlock-console  
Service to route to.

**Target Port \***  
8083 → 8083 (TCP)  
Target port for traffic.

**Security**  
☒ Secure route  
Routes can be secured using several TLS termination types for serving certificates.

**TLS Termination \***  
Passthrough

**Insecure Traffic**  
Redirect  
Policy for traffic on insecure schemes like HTTP.

Create Cancel

- Navigate to the URL under 'Location' on the right side of the OpenShift Console.
- Create an admin account login.



Create an administrator account

admin

Pal0Alt0@123

☒ Show password

Create account

- Enter your license key when prompted.

Manage / System

License

Twistlock license

*i* Update your Twistlock license.

Insert license key here

Register

### License Key:

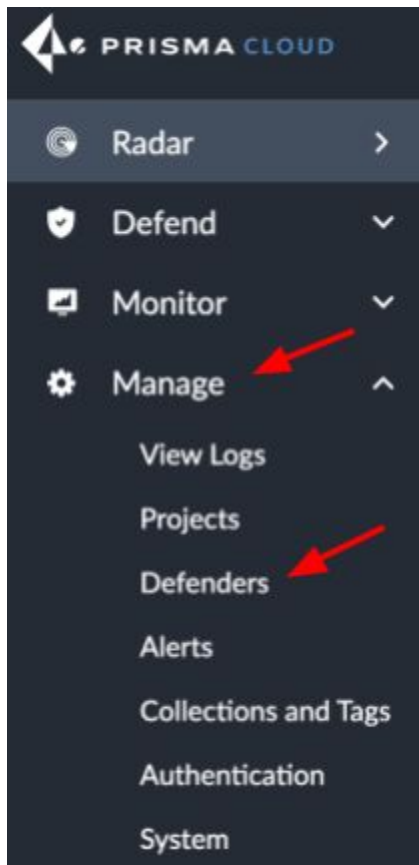
Z3tYy2wEjIXzSm2XmYUq56eGRW4cACsDixbYQJieHpEh8806W2ldZENJddZhTnUKzloRObl9Vs  
 4lp+R3MVD2tCBv91yrMpDyw+XjFBqR/Wp8yXEh1DUjR042oisC9ztaSy98f0G1lumcDa7W1RgeT  
 X1Ld9RRnxELIve9JpcsTRStjqJDb1+iiZtvEv4i+p1xG7C2UI6WRzgCMFQ3QvrhBeA9UICCP+f3Rrf  
 FsuvIYVoxhzARlsc4cVAuS9STnuzdj+6vvlQcZtnkCyYmRt4qMPf1y93JShNbu5cZ2rgtYCzqii8WH  
 RnAxrMfSS1yuyS6GgawUxbO4vVzzO66zVPu110ickihspiD0/a4TcenJwupDqo0uyUKHIQNBj6dU  
 gvt3XbBJOGEIqdmYcTSGHh/cxgt0FaCkCwSfHwijbgrwAZixeK0xkLCQgXdTKGnyvoaE7xEejXC  
 16UgqHLA/n9LW3jaqMVMcPa7oSiCOncXcYjpsop+/CljXQhpuKhSmw+xkkpXTPjuch8LJrWBtnqla  
 WjMJQOxrFrlxKrB9FHw2RzluXVjFf9wSUjPHHrFr3prqMmEKIIBqvLsHoR5uJ+85mfFJT6kRlnUcll  
 oy4dDFjRZc1Cp2Q0m8HI6b9ZVWcScjBJSUpXYxu9ileqa2D9rQQupTe64wbbYBKU9CerkGmQ=

Congratulations, You have now installed Prisma Cloud Compute onto an OpenShift Cluster!

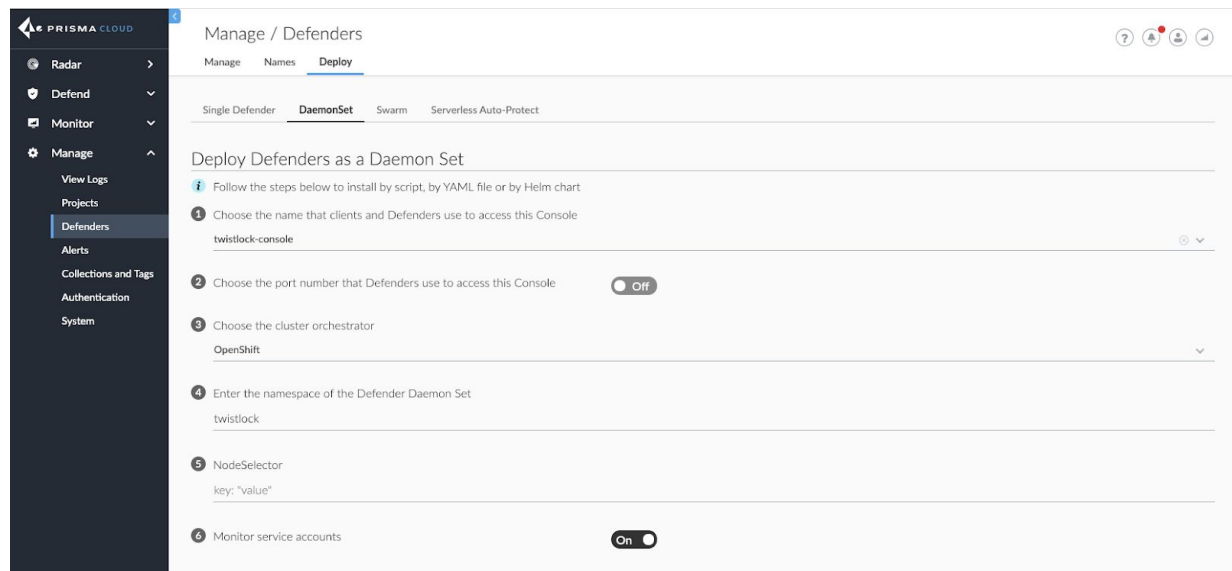
# Installing Defender Agents to Workloads

Now we will install the Prisma Cloud Compute Defender Daemonset. The Defender is a software module that will be installed to monitor every Container. Defender communicates with the Prisma Cloud Compute Console using Transport Layer Security (TLS). You will update the list of identifiers in Console's certificate that the Defenders will use to validate the Console's identity. This will change your current browser certificate and force you to log back in to the Prisma Cloud Compute Console.

- Install Prisma Cloud Compute Defender DaemonSet. From the Prisma Cloud Compute Console go to Manage > Defenders.



- **Deploy > DaemonSet.**



Manage / Defenders

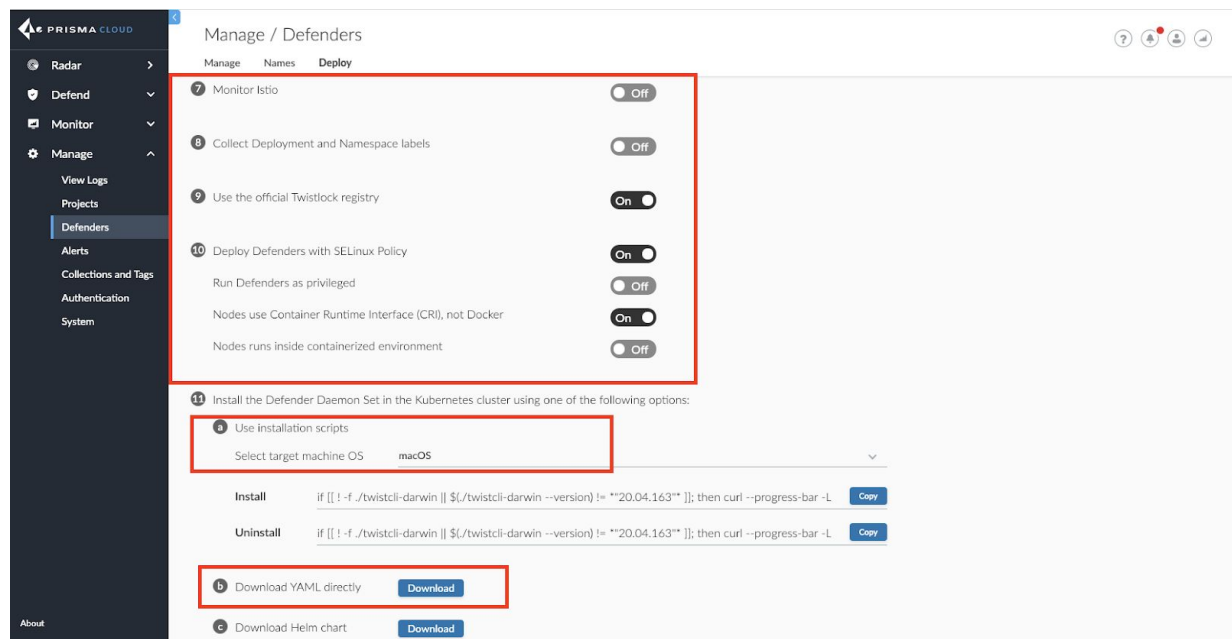
Manage Names Deploy

Single Defender **DaemonSet** Swarm Serverless Auto-Protect

### Deploy Defenders as a Daemon Set

Follow the steps below to install by script, by YAML file or by Helm chart

- Choose the name that clients and Defenders use to access this Console  
twistlock-console
- Choose the port number that Defenders use to access this Console ☐ Off
- Choose the cluster orchestrator  
OpenShift
- Enter the namespace of the Defender Daemon Set  
twistlock
- NodeSelector  
key: "value"
- Monitor service accounts ☒ On



Manage / Defenders

Manage Names Deploy

- Monitor Istio ☐ Off
- Collect Deployment and Namespace labels ☐ Off
- Use the official Twistlock registry ☒ On
- Deploy Defenders with SELinux Policy ☒ On
  - Run Defenders as privileged ☐ Off
  - Nodes use Container Runtime Interface (CRI), not Docker ☒ On
  - Nodes runs inside containerized environment ☐ Off
- Install the Defender Daemon Set in the Kubernetes cluster using one of the following options:
  - Use installation scripts
 

Select target machine OS: macOS

**Install** `if [[ ! -f ./twistcli-darwin || $(./twistcli-darwin --version) != "20.04.163" ]]; then curl --progress-bar -L` [Copy](#)

**Uninstall** `if [[ ! -f ./twistcli-darwin || $(./twistcli-darwin --version) != "20.04.163" ]]; then curl --progress-bar -L` [Copy](#)
  - Download YAML directly** [Download](#)
  - Download Helm chart [Download](#)

- The client defender name should be **"twistlock-console"**. The defender is installed as a DaemonSet, which ensures that an instance of defender runs on every node in the cluster.
- On the same page scroll down the page. Select the target machine OS and choose your device's OS. Then select **Download** YAML directly.
- Download the defender daemonset YAML file directly from the console.
- Move/copy the daemonset.yaml file to working directory.
- Ensure you the 'oc' client is set to the 'twistlock' namespace.  
oc project twistlock



- Deploy the defender daemonset from the directory to which it downloaded (or move file).  
\$ oc create -f ./daemonset.yaml
- Expected output:  
clusterrole.rbac.authorization.k8s.io/twistlock-view created  
clusterrolebinding.rbac.authorization.k8s.io/twistlock-view-binding created  
securitycontextconstraints.security.openshift.io/twistlock-scc created  
secret/twistlock-secrets created  
serviceaccount/twistlock-service created  
daemonset.apps/twistlock-defender-ds created  
service/defender created
- Check the OpenShift console to see that pods have been deployed. **Workloads > Pods**.

Project: all projects ▾

### Pods

Filter ▾ Name ▾ twist

Name twist ✕ Clear all filters

Name ↑	Namespace ↑	Status ↑	Ready ↑	Restarts ↑	Owner ↑	Mem
network-metrics-daemon-tm57h	openshift-multus	Running	2/2	0	network-metrics-daemon	50.6 I
twistlock-console-7cf6869444-84dcb	twistlock	Running	1/1	0	twistlock-console-7cf6869444	552.7
twistlock-defender-ds-knr19	twistlock	Running	1/1	0	twistlock-defender-ds	34.9 I
twistlock-defender-ds-wncbk	twistlock	Running	1/1	0	twistlock-defender-ds	34.7 I

- (Optional) Also check daemonsets via oc cli  
\$ oc get ds -n twistlock
- The Defender Daemon Set install script creates and deploys the Prisma Cloud Compute defender.yaml file. The script will direct the Defender to connect to the Prisma Cloud Compute console using the service account the script creates. It may take a minute for the defender.yaml to be fully deployed.
- 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:

Manage / Defenders

Manage Names Deploy

Defenders DaemonSets

### Manage deployed Defenders

*i* Defenders enforce the policies created in Console. Install Defender on each host you want Prisma Cloud to protect. [Advanced settings](#)

Filter Defenders by keywords and attributes 2 total entries

Host	Version	Cluster	Type	Listener type	Roles
ip-10-0-149-75	20.12.535	cluster-7801	Daemon Set CRI on Linux	None	
ip-10-0-134-65	20.12.535	cluster-7801	Daemon Set CRI on Linux	None	

- Next, 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.

PRISMA CLOUD

Color by: Vulnerabilities

Filter by collection or namespace

Search by image name

Deployed Defenders

- Container Defenders
- Host Defenders
- Serverless Defenders
- App embedded Defenders

Number of Incidents

Last 7 days

Not enough historical data to graph yet

Compliance Vulnerabilities

Impacted images

Impacted containers

Impacted hosts

Impacted functions

5 hours since last Intelligence Stream update

Gathering initial stats, which takes about a day

openshift-dns

openshift-sd

openshift-image-registry

openshift-multus

twistlock

openshift-monitoring

openshift-machine-config-operator

terminal

openshift-ingress

openshift-console

demo

openshift-cluster-node-tuning-operator

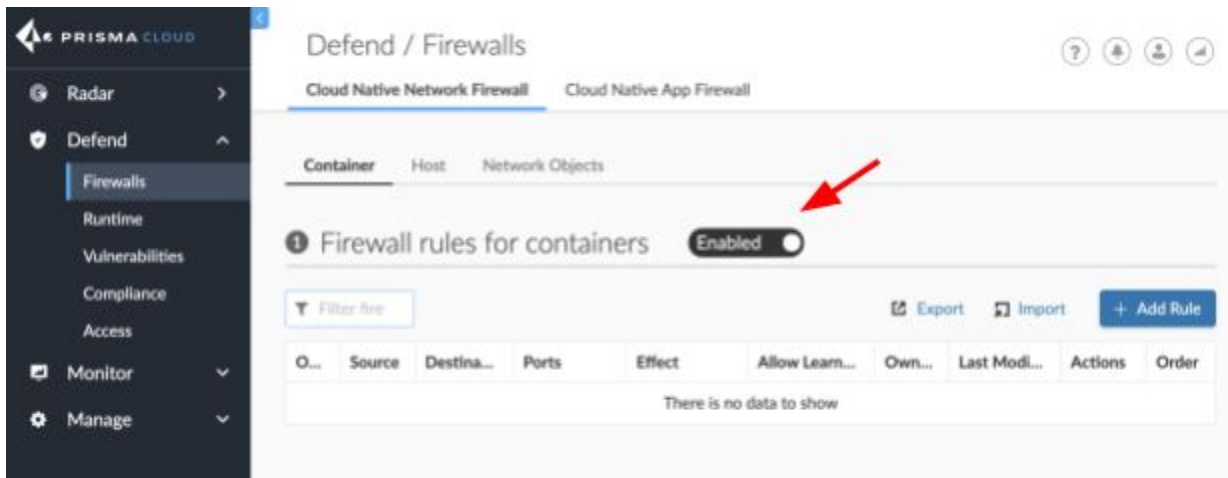
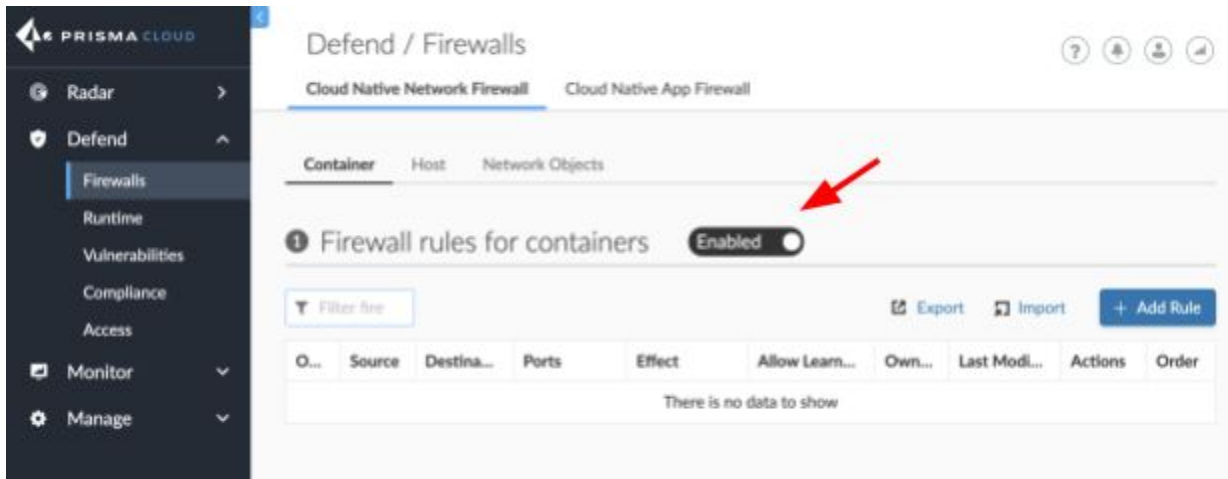
openshift-kube-storage-version-migrator

openshift-cluster-csi-drivers

openshift-marketplace

Containers

- Enable the Cloud Native Network Firewall (CNNF) within Prisma Cloud Compute. Go to **Defend -> Firewalls** and click on the flip switch to enable '**Firewall rules for containers**'.



- Click **Relearn**.

---

# Deploy the Sock Shop webpage

---

## Deploy the Sock Shop webpage

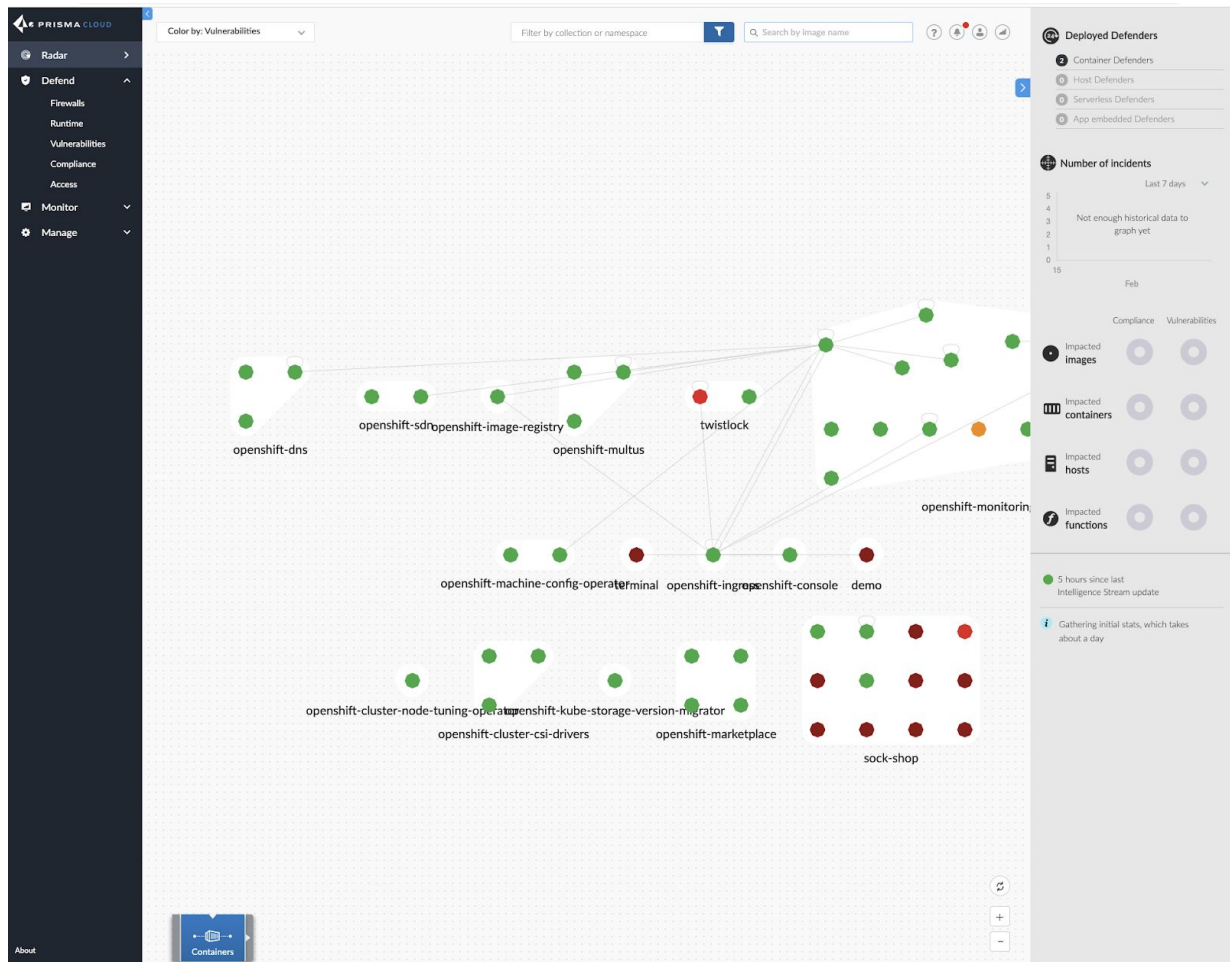
- Latest sock shop:  
<https://github.com/mosuke5/microservices-demo-openshift>  

```
$ git clone https://github.com/mosuke5/microservices-demo-openshift.git  
$ cd microservices-demo-openshift  
$ oc new-project sock-shop  
$ oc apply -f complete-demo.yaml  
$ oc expose service/front-end  
$ oc get route
```
- In the OpenShift console, navigate to **Networking > Routes**, to view the route you just created. Follow the link under the **Location** field.

The screenshot shows the Red Hat OpenShift Container Platform console interface. The left sidebar contains navigation menus for Administrator, Home, Operators, Workloads, and Networking. The 'Routes' page is selected under the Networking menu. The main content area displays the 'Routes' for the 'Project: sock-shop'. A table lists the routes, with one entry visible: 'front-end' with a status of 'Accepted'. The 'Location' field contains a URL: 'http://front-end-sock-shop.apps.cluster-paloalto-c13d.paloalto-c13d.example.opentlc.com'. A 'Create Route' button is located in the top right corner of the main content area.

Name	Status	Location	Service
front-end	Accepted	<a href="http://front-end-sock-shop.apps.cluster-paloalto-c13d.paloalto-c13d.example.opentlc.com">http://front-end-sock-shop.apps.cluster-paloalto-c13d.paloalto-c13d.example.opentlc.com</a>	front-end

- In the Prisma Cloud Compute console, navigate to **Radar**, to view the containers created for the sock-shop. Follow the link under the **Location** field.



**Congratulations!** You've successfully deployed the sock-shop microservices which are being protected by defenders.

---

# Malware Demonstration

---

Without secure hosts, you cannot have secure containers. Host machines are a critical component in the container environment, and they must be secured with the same care as containers. **Runtime defense** employs machine learning to build models of known good activity so that anomalous runtime behavior can quickly be detected. Prisma Cloud applies our model-based approach to runtime defense to protect the hosts in your environment.

## Host Models

Whereas container runtime protection creates models for images, host runtime protection creates models for systemd services. When Defender is installed on a host, it automatically creates runtime models for every service and container that runs on the host.

When a new service is detected, Prisma Cloud creates a new model for it, and puts the model into **learning mode**. Baseline models are provided for common Linux services, such as httpd (Apache) and sshd. All services, whether they have baseline models or not, are subject to a learning period before their models are fully activated.

There is one model per service, and models are global. A model for one host service is shared and reused amongst all hosts. For example, consider a service `foo`, which runs on host-A and host-B. If `foo` acquires the DOCKER capability on host-A and the USERS\_ADMIN capability on host-B, then the model, shared across both hosts, would enumerate both capabilities.

You can view all runtime models for your host service in Console under **Monitor > Runtime > Host Models**. The table on this page has two pivot points:

- **Host** – Lists all the hosts in your environment. Click on a host to see the services on the host and their associated models.
- **Service** – Lists all the services in your environment. You can review the state of the models, and either relearn a model, or activate a manual learning mode for a model. Clicking on a service shows you the capabilities that have been allowed for it.

## Capabilities

Host models map capabilities to services. Capabilities are Prisma Cloud-curated units of process and file system actions that express the things that services routinely need to do. They can be independently enabled or disabled on a per-service basis, and provide fine-grained control over what a service can and cannot do.

Capabilities encapsulate the intersection of what services need to do and what attackers want to do. Capabilities were borne from the following observations:

- Attacker objectives are well known. They need to gain a footprint on the host, establish persistence, elevate permissions, and so on.

- The path to achieving these objectives is also well known. Attackers must use specific utilities and manipulate specific files in order to advance their position. In many cases, these are the same utilities and files that legitimate services need to use.

By selectively assigning capabilities to services so that they can do their job, but nothing more, Prisma Cloud can limit what an attacker can do when they hijack a service and try to exploit it to run in non-legitimate ways.

Capabilities have two key traits:

- Processes they can run.
- Files they can access.

For example, the USER\_ADMIN capability permits user management. The following process and configuration files fall under its purview:

Processes:

- adduser
- useradd

file system:

- /etc/group
- /etc/passwd

## List of capabilities

The following capabilities can be applied to a service model. The last column in the table provides representative examples of files and processes that are associated with each capability.

CAPABILITY	DESCRIPTION	EXAMPLE
SYSTEM_LOGS	Reading and changing system logs	Example files: <ul style="list-style-type: none"> <li>• /var/log/syslog (System log)</li> <li>• /var/log/audit/audit.log (Audit daemon log)</li> <li>• /var/log/auth.log (Authentication logs)</li> <li>• /var/log/kern.log (Kernel logs)</li> </ul>
DOCKER_SOCKET	Access to Docker socket	Example files: <ul style="list-style-type: none"> <li>• /var/run/docker.sock</li> </ul>



DOCKER	Changing Docker configuration	<p>Example files:</p> <ul style="list-style-type: none"> <li>• <code>/etc/docker</code></li> </ul> <p>Example processes:</p> <ul style="list-style-type: none"> <li>• <code>docker-containerd</code></li> <li>• <code>docker-containerd-current</code></li> <li>• <code>docker-proxy</code></li> </ul>
K8S	Changing Kubernetes configuration	<p>Example files:</p> <ul style="list-style-type: none"> <li>• <code>/etc/kubernetes</code></li> <li>• <code>~/.kube</code></li> </ul> <p>Example processes:</p> <ul style="list-style-type: none"> <li>• <code>kubelet</code></li> <li>• <code>kubectrl</code></li> </ul>
OPENSIFT	Changing OpenShift configuration	<p>Example files:</p> <ul style="list-style-type: none"> <li>• <code>/usr/share/openshift</code></li> </ul> <p>Example processes:</p> <ul style="list-style-type: none"> <li>• <code>oc</code></li> <li>• <code>openshift</code></li> </ul>
USERS	Reading users and groups	<p>Example files:</p> <ul style="list-style-type: none"> <li>• <code>/etc/passwd</code> (User database)</li> <li>• <code>/etc/group</code> (User groups)</li> </ul>
USERS_ADMIN	Changing users and groups	All files in the USERS capability, but these files are monitored for write activity.

SSH	Changing sshd configuration	<p>Example files:</p> <ul style="list-style-type: none"> <li>• /etc/ssh</li> <li>• /etc/ssh/sshd_config</li> <li>• SSH keys under ~/.ssh</li> </ul> <p>Example processes:</p> <ul style="list-style-type: none"> <li>• ssh</li> <li>• sshd</li> <li>• vpn</li> </ul>
SHELL	Running a shell	<p>Example files:</p> <ul style="list-style-type: none"> <li>• /etc/profile</li> </ul> <p>Processes:</p> <ul style="list-style-type: none"> <li>• bash</li> <li>• ash</li> <li>• busybox</li> </ul>
NET	Changing network configuration	<p>Example processes:</p> <ul style="list-style-type: none"> <li>• iptables</li> <li>• socat</li> <li>• nc</li> </ul>
HOSTS_CONFIG	Changing host configuration	<p>Example files:</p> <p>/etc/hosts/etc/resolv.conf</p>
GLOUD	Changing Google Cloud configuration	
INSTALLER	Installing software	<p>Example processes:</p> <ul style="list-style-type: none"> <li>• apt</li> <li>• rpm</li> <li>• Writing any binary to disk.</li> </ul>
SERVICE_OPERATOR	Managing services	<p>Example processes:</p> <ul style="list-style-type: none"> <li>• systemd</li> <li>• service</li> <li>• systemctl</li> </ul>

PRIVILEGED\_PROCESSES    Running privileged processes

## Violations

When a service requests a capability that isn't in its model, Prisma Cloud generates an audit. Audits can be viewed under **Monitor > Events > Host Audits**.

Consider an HTTP server that executed the `useradd` command. There should be no reason for an HTTP server to execute the `useradd` command. The `useradd` command is part of the `USERS_ADMIN` capability, and the `http-server` service's model won't have this capability. As a result, you get an audit:

```
Service http-server attempted to obtain capability USERS_ADMIN by
executing /usr/sbin/useradd
```

You can create rules that prevent a service from running a process or accessing a file that is part of a capability (the default action is alert). Create a new host runtime rule, select the **Capabilities** tab, and set the action for the capability to **Prevent**. In the **General** tab, be sure to scope the rule to a specific service. Otherwise, you will disrupt other services.

With this rule in place, further attempts to exploit `http-server` are blocked.

## Enabling host runtime protection

Runtime protection for hosts is enabled by default. When Defender is installed on a host, it automatically starts building runtime models for all services. Prisma Cloud ships with a default rule named `Default - alert on suspicious runtime behavior`, which alerts on all anomalous activity. To see the rule, open Console, then go to **Defend > Runtime > Host Policy**.

As part of the default rule, Prisma Cloud Advanced Threat Protection (TATP) is enabled. TATP supplements runtime protection by alerting you when:

- Malware is found anywhere on the host file system.
- Connections are made to banned IP addresses.

Create new rules to enhance host protection.

Do not allow or deny capabilities globally by setting the host filters to a wildcard because it can interrupt the execution of legitimate services.

## Anomalous app detection

Prisma Cloud learns the normal set of apps running on your hosts (a so-called app baseline), and automatically identifies apps added abnormally. After all running apps complete the learning phase, any subsequently started apps generate a one time audit.

## Log inspection

Prisma Cloud lets you collect and analyze operating systems and application logs for security events. For each inspection rule, specify the log file to parse and any number of inspection expressions. Inspection expressions support the **RE2 regular expression syntax**.

A number of predefined rules are provided for apps such as sshd, mongod, and nginx.

## Networking

Prisma Cloud lets you secure host networking. You can filter DNS traffic and alert on inbound and outbound connections.

### DNS

When DNS monitoring is enabled, Prisma Cloud filters DNS lookups. By default, DNS monitoring is disabled. Dangerous domains are detected as follows:

- Prisma Cloud Intelligence Stream -- Prisma Cloud's threat feed contains a list of known bad domains.
- Explicit allow and deny lists – Host runtime rules let you augment the Prisma Cloud's Intelligence Stream data with your own lists of known good and bad domains.

In your runtime rules, set **Effect** to configure how Defender handles DNS lookups from containers:

- **Disable** – DNS monitoring is disabled.
- **Alert** – DNS monitoring is enabled. Anomalous activity generates audits.
- **Prevent** – DNS monitoring is enabled. Anomalous activity generates audits. Anomalous DNS lookups are dropped.

### IP connectivity

You can raise alerts when inbound or outbound connections are established. Specify inbound ports, and outbound IPs and ports.

Outbound connections are event-driven, which means that as soon as a process attempts to establish a connection, you'll be notified. Prisma Cloud polls inbound connections, which means you'll be notified periodically, and not necessarily the moment an inbound connection is established.

### Access logging

Set up rules to audit **host events**.

### File integrity management (FIM)

Changes to critical files can reduce your overall security posture, and they can be the first indicator of an attack in progress. Prisma Cloud FIM continually watches the files and directories in your monitoring profile for changes. You can configure to FIM to detect:

- Reads or writes to sensitive files, such as certificates, secrets, and configuration files.
- Binaries written to the file system.
- Abnormally installed software. For example, files written to a file system by programs other than apt-get.

A monitoring profile consists of rules, where each rule specifies the path to monitor, the file operation, and exceptions.

Add a new file integrity rule

Path	<input type="text" value="e.g. /etc"/>
Recursive	<input type="checkbox"/> Monitor subdirectories
Write	<input type="checkbox"/> Monitor write operations
Read	<input type="checkbox"/> Monitor read operations
Metadata	<input type="checkbox"/> Monitor changes to metadata
Allowed processes	<input type="text" value="Process name (e.g., bash)"/>
Excluded file patterns	<input type="text" value="Filename pattern (e.g., *.k)"/>

Cancel Add File Integrity Rule

The file operations supported are:

- Writes to files or directories. When you specify a directory, recursive monitoring is supported.
- Reads. When you specify a directory, recursive monitoring isn't supported.
- Attribute changes. The attributes watched are permissions, ownership, timestamps, and links. When you specify a directory, recursive monitoring isn't supported.

## End of Workshop!

### Links/contacts for more information:

Prisma Cloud compute administrators guide for OpenShift:

[https://docs.paloaltonetworks.com/prisma/prisma-cloud/20-04/prisma-cloud-compute-edition-admin/install/install\\_openshift.html](https://docs.paloaltonetworks.com/prisma/prisma-cloud/20-04/prisma-cloud-compute-edition-admin/install/install_openshift.html)

Create OpenShift project for Prisma Cloud:

[https://docs.twistlock.com/docs/compute\\_edition/install/install\\_openshift\\_4.html#download-the-prisma-cloud-software](https://docs.twistlock.com/docs/compute_edition/install/install_openshift_4.html#download-the-prisma-cloud-software)

Security for Red Hat OpenShift:

<https://www.paloaltonetworks.com/prisma/environments/red-hat-openshift>



Copyright © 2020 Red Hat, Inc. Red Hat, Red Hat Enterprise Linux, the Red Hat logo, and JBoss are trademarks or registered trademarks of Red Hat, Inc. or its subsidiaries in the United States and other countries. Linux is the registered trademark of Linus Torvalds in the U.S. and other countries.