

Red Hat Advanced Cluster Security

With Kasten K10

Quick Start Guide January 2024

Red Hat Advanced Cluster Security with Kasten K10

Below is an example guide of how someone may be able to deploy Red Hat Advanced Cluster Security in conjunction with Kasten K10 for Kubernetes in an OCP cluster to ensure a security posture fit for your deployment that correlates and heightens events that are associated with custom Kasten K10 resources.

Setting up RHACS and Kasten K10 on an OCP Cluster

In this example, we'll deploy a Red Hat OpenShift cluster on AWS. Once our cluster is ready, we can install Red Hat Advanced Cluster Security either via Helm or the OpenShift Operator. Let's go ahead and install via Helm by getting the chart:

helm repo add rhacs https://mirror.openshift.com/pub/rhacs/charts/

We need to ensure we have access to the Red Hat registry at <u>https://catalog.redhat.com/</u> before proceeding or else the image pull will fail. For example, if using docker you can run the following command with your Red Hat username and password:

docker login -u "\$username" -p "\$password"

Next, let's create a stackrox namespace and secret to hold our Red Hat registry credentials:

```
kubectl create ns stackrox
kubectl create secret generic redhatcreds -n stackrox \
    --from-file=.dockerconfigjson="$HOME/.docker/config.json" \
    --type=kubernetes.io/dockerconfigjson
```

Now, let's install the central services with port forwarding (you can also use an exposed route or a load balancer, see <u>here</u> for help):

```
helm install -n stackrox \
    --create-namespace stackrox-central-services rhacs/central-services \
    --set imagePullSecrets.useExisting=redhatcreds
```

Once installed you should see an admin password listed to be used with the Red Hat Advanced Cluster Security console. Copy this as we'll need it.

At this point pods for the centralized services should be creating. Once these are created, make sure to port forward the central services:

kubectl -n stackrox port-forward svc/central 18443:443

Now open your browser and go to https://localhost:18443. You should be prompted for a username and password; use admin and the password copied after helm installed the centralized services.

ed Hat dvanced Cluster Securi
or Kubernetes

The dashboard should show empty with no clusters monitored since we haven't installed the secured cluster services yet.

Red Hat Advanced Cluster Sec for Kubernetes	urity	Q Search 🛓 CLI 🔅 4r 😧 🚽
Dashboard	0 0 0 0 0 0 Clusters Nodes Violations Deployments Images	0 Last updated 12/23/2022 at 9:27 AM
Network Graph	Dashboard	
Violations	Review security metrics across all or select resources	Resources: All clusters All namespaces
Compliance		
Vulnerability Management >	0 policy violations by severity	all Images at most risk Options View all
Configuration Management	0 0 0 0 Low Medium High Critical	
Risk		Q
Platform Configuration	Most recent violations with critical severity	No data was found in the selected resources.
	Q	
	No data was found in the selected resources.	
	Deployments at most risk View	all O Aging images Options - View all

To do this, navigate to "Platform Configuration" in the left sidebar, and select "Integrations"; scroll to the bottom and click "StackRox API Token":

Red Hat Advanced Cluster Secu for Kubernetes	rity		م	Search 🛓 CLI	× 4	0
hboard						
twork Graph	Backup Integrations					
iolations	amazon S3	O Google Cloud Platform				
ulnerability Management	Amazon S3	Google Cloud Storage				
isk						
latform Configuration 🛛 🗸	Authentication Tokens					
Clusters Policy Management Integrations	StackRox	StackRox				
Access Control System Configuration System Health	API Token	Cluster Init Bundle				

Create an API Token called "Admin" with the "Admin" Role and copy the generated token

Red Hat Advanced Cluster Sector for Kubernetes	ırity	Q Search	🛃 CLI	÷,	v 0	ad
Dashboard	Integrations > API Token > Creste Integration					
Network Graph	Configure API Token Integration					
Violations						
Compliance	Token name *					
Vulnerability Management >	admin					
Configuration Management	Admin -					
Risk						
Platform Configuration 🛛 🗸						
Clusters						
Policy Management						
Integrations						
Access Control						
System Configuration						
System Health	Generate Cancel					

Next, we need to create the init bundle to allow the cluster to authenticate with Central. You'll need the roxctl cli and for a Mac can be installed as follows:

```
curl -0
https://mirror.openshift.com/pub/rhacs/assets/3.70.2/bin/Darwin/roxctl
xattr -c roxctl
chmod +x roxctl
sudo mv roxtl /usr/local/bin
```

Set the ROX API TOKEN environment variable with the copied API token:

export ROX API TOKEN=\${API TOKEN}

Set the ROX CENTRAL ADDRESS environment variable:

export ROX CENTRAL ADDRESS=https://localhost:18443

Now, create the init bundle:

```
roxctl -e "$ROX_CENTRAL_ADDRESS" central init-bundles generate acs-test --
output cluster_init_bundle.yaml
```

With the init bundle created, install the secure cluster services on each cluster you want monitored:

```
helm install -n stackrox \
    --create-namespace stackrox-secured-cluster-services rhacs/secured-cluster-
services \
    -f PATH_TO_INIT_BUNDLE/cluster_init_bundle.yaml \
    --set imagePullSecrets.useExisting=redhatcreds \
    --set clusterName=acs-test \
```

--set centralEndpoint=central.stackrox:443

This will create the admission-control, collector, scanner-db, and sensor pods. Once these are all running Red Hat Advanced Cluster Security is ready to use.

Lastly, install the latest version of Kasten K10 through the OperatorHub, by following the directions found in the <u>Kasten K10 documentation</u>.

Viewing Kasten K10 Audit Logs In OCP Cluster

The audit policy in a Red Hat OpenShift cluster is customizable in the sense that it allows you to select from three different settings which correlate to the Metadata, Request, and RequestReceived levels that the kube audit logs at.

Running the following command brings up the config.openshift.io API group's APIServer resource which can be edited:

oc edit apiserver cluster

Edit the spec.audit.profile field to one of Default, WriteRequestBodies, AllRequestBodies to have increasingly more information present in the audit event that's logged.

Let's find the nodes and location of the audit logs by running:

oc adm -role=master --path=kube-apiserver/audit.log

Depending on the number of nodes, multiple nodes and audit logs should come up. Select the node name and location and run the following to see the audit logs:

```
oc adm node-logs ip-10-0-255-61.us-west-1.compute.internal --path=kube-
apiserver/audit.log
```

To test this, let's run the following command to list and describe the k10MasterKey:

kubectl get passkeys
kubectl describe k10MasterKey

Let's search through the logs to find this audit event (you may need to look through multiple nodes and locations to find this):

```
oc adm node-logs ip-10-0-255-61.us-west-1.compute.internal --path=kube-
apiserver/audit.log | jq 'select(.requestURI |
startswith("/apis/vault.kio.kasten.io/v1alpha1"))'
```

We then see the following audit event:

```
{
 "kind": "Event",
 "apiVersion": "audit.k8s.io/v1",
 "level": "Metadata",
 "auditID": "11aa507b-e3a6-4be6-b5d7-458f4d69dc37",
 "stage": "ResponseComplete",
 "requestURI": "/apis/vault.kio.kasten.io/v1alpha1/passkeys/k10MasterKey",
 "verb": "get",
 "user": {
    "username": "system:admin",
    "groups": [
     "system:masters",
     "system:authenticated"
   1
 },
 "sourceIPs": [
   "10.0.117.141"
 ],
 "userAgent": "kubectl/v1.24.0 (linux/arm64) kubernetes/4ce5a89",
 "objectRef": {
   "resource": "passkeys",
   "name": "k10MasterKey",
   "apiGroup": "vault.kio.kasten.io",
   "apiVersion": "vlalpha1"
 },
 "responseStatus": {
   "metadata": {},
   "code": 200
 },
 "requestReceivedTimestamp": "2022-12-23T20:01:31.6030202",
 "stageTimestamp": "2022-12-23T20:01:31.605900Z",
 "annotations": {
    "authorization.k8s.io/decision": "allow",
    "authorization.k8s.io/reason": ""
 }
}
```

Actions against Kasten K10's custom resources get processed by the *kube-apiserver* which then logs the audit event based on the audit policy. These can be from custom resources created by CRD's or the Aggregated API; Kasten K10's cloud native architecture natively allows for the *kube audit* to be able to audit interactions with Kasten K10, ensuring protection against unauthorized access and misuse of Kasten K10's underlying Kubernetes structure.

Using RHACS Console to Monitor Kasten K10

With Kasten K10 installed in our cluster, we can view the Red Hat Advanced Cluster Security dashboard for the kasten-io namespace to get high-level information as to the security of Kasten K10; Select acs-test from the list of clusters in the top right corner, and then choose kasten-io from the list of namespaces.

Here we see policy violations by severity, active images at risk, deployments at most risk, active aging images, policy violations by category, and compliance by standard.

Red Hat Advanced Cluster Securit for Kubernetes	У				Q Search	🛓 CLI 🔅	∿ ? (ad)
Dashboard	36 policy violation	as by severity	View all	Active images at most	risk	Option	ns 🔻 View all
Network Graph	21	15 0	0	Images	Risk priority	Critical CVEs	Important CVEs
Violations	Low	Medium High	Critical	rich-access-174020/emissary	6	V 0 fixable	💔 O fixable
				kasten/k10-operator	6	🛡 0 fixable	💔 O fixable
Compliance	Most recent violations	with critical severity		rich-access-174020/cephtool	7	🛡 O fixable	💔 O fixable
Vulnerability Management				rich-access-174020/frontend	9	U fixable	💙 0 fixable
		Q		rich-access-174020/kanister	9	🛡 0 fixable	😈 O fixable
Configuration Management	No da	ata was found in the selected reso	urces.	rich-access-174020/admin	9	😈 0 fixable	U fixable
Risk							
Platform Configuration	Deployments at m	nost risk	View all	0 Active aging images		Option	ns ▼ View all
	Deployment	Resource location	Risk priority				
	executor-svc	in "eliott / kasten-io"	59				
	state-svc	in "eliott / kasten-io"	63		0		
	logging-svc	in "eliott / kasten-io"	64	2			
	gateway	in "eliott / kasten-io"	65	No data wa	as tound in the	selected resourc	es.
	crypto-svc	in "eliott / kasten-io"	66				
	kanister-svc	in "eliott / kasten-io"	67				

We can follow links to get more information about any of the data presented here.

If we select "Compliance" from the left side bar, we can scan our environment to run a compliance check.

Red Hat Advanced Cluster Sec for Kubernetes	urity		Q Search 🛓 CLI 🔅 🍾 🕢			
Dashboard	COMPLIANCE Dashboard	0 CLUSTERS 0 NAMESPACES 0 NODES (Scanned) (Scanned) (Scanned)	D DEPLOYMENTS (Scanned)			
Network Graph						
Violations	PASSING STANDARDS ALROSS CLUSTERS	PASSING STANDARUS OF LLUSTER	PASSING STANUARUS ALRUSS NAMESPALES			
Compliance						
Vulnerability Management 🔹						
Configuration Management	No data available. Please run a scan.	No data available. Please run a scan.	No data available. Please run a scan.			
Risk						
Platform Configuration						
	PASSING STANDARDS ACROSS NODES	CIS DOCKER V1.2.0 COMPLIANCE	CIS KUBERNETES V1.5 COMPLIANCE			
	No data available. Please run a scan.	No data available. Please run a scan.	No data available. Please run a scan.			

From here we can deep-dive different standards such as <u>NIST SP 800-190</u> (Application Container Security Guide) and <u>NIST SP 800-53</u> (Security and Privacy Controls for Information Systems and Organizations).

We can generate the network graph for the kasten-io namespace by selecting "Network Graph" from the left sidebar and then selecting this namespace. If we select the gateway pod, we can see the network connections and any anomalous flows Red Hat Advanced Cluster Security finds:



Since Red Hat Advanced Cluster Security comes with a lot of security policies by default, such as monitoring <code>kubectl port-forward...</code> and <code>kubectl exec...</code>, if any of these actions happen on a K10 pod, a violation will show up. These violations are all shown by clicking on "Violations" in the left sidebar.

Red Hat Advanced Cluster Secu for Kubernetes	rity					٩	Search 🛃 C	⊔ ◊ ᠰ	? ad
Dashboard	Violations								
Network Graph	V Namespace: X kasten-jo X								xv
Violations									
Compliance	18 results found Row Actions	*				1	-18 of 18 👻	« < 1 of1	
Vulnerability Management >	Policy	Entity	Туре	Enforced 1	Severity 1	Categories 1	Lifecycle	Time I	
Configuration Management	 No resource requests or limits specified 	logging-svc in "acs/kasten-io"	deployment	No	Medium	Multiple	Deploy	12/23/2022 4:41:52PM	1
Risk	No resource requests or limits specified	controllermanager-	deployment	No	Medium	Multiple	Deploy	12/23/2022 4/41-57PM	I
Platform Configuration	specineu	in "acs/kasten-io"						4.41.37 FM	
	No resource requests or limits specified	auth-svc in "acs/kasten-io"	deployment	No	Medium	Multiple	Deploy	12/23/2022 4:41:59PM	I
	No resource requests or limits specified	dashboardbff-svc in "acs/kasten-io"	deployment	No	Medium	Multiple	Deploy	12/23/2022 4:41:59PM	I
	No resource requests or limits specified	jobs-svc in "acs/kasten-io"	deployment	No	Medium	Multiple	Deploy	12/23/2022 4:42:00PM	1
	No resource requests or limits specified	prometheus-server in "acs/kasten-io"	deployment	No	Medium	Multiple	Deploy	12/23/2022 4:42:10PM	:

Conclusion

Red Hat Advanced Cluster Security in conjunction with Kasten K10 allows for an enhanced security posture to further protect Kubernetes data from unknown attacks from malicious users.

Footnotes

1. You cannot install Red Hat Advanced Cluster Security in a local cluster such as with K3D or Minikube, or in a Lima VM running x86 ubuntu with K3S installed; it must be in one of the listed managed Kubernetes services.