Solution Brief

aqua

Securing your K8s Infrastructure with Aqua KSPM

As cloud native adoption continues to grow, so does the need to control the many configuration options in the infrastructure that runs cloud native applications and ensure they don't introduce security risks.

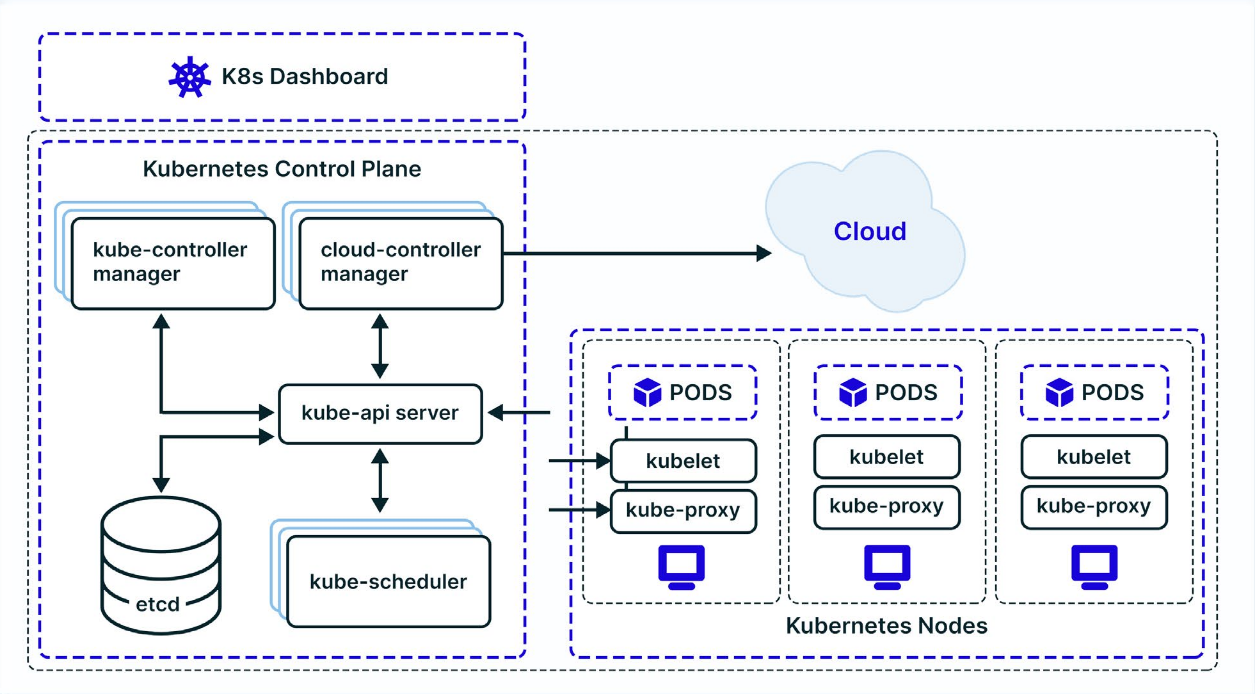
Just as the security of public cloud services can be automated using Cloud Security Posture Management (CSPM), ensuring that configurations remain secure and avoiding human error, Kubernetes is no exception – it is a complex system with many moving parts and requires automation and the consistent application of security best practices, especially as the scale and diversity of use cases increase.

As the de facto standard in container orchestration, Kubernetes has also become an attractive target for cybercriminals and other threat actors. We have seen organized attacks that target and exploit weaknesses in Kubernetes clusters. Automated configuration and posture checks, similar to those done as part of CSPM, are therefore critical for organizations that use Kubernetes in production.

Navigating the complexities of Kubernetes security

First, as a system with many different services and components, the potential attack surface exposed in Kubernetes can be quite broad. Narrowing that attack surface is challenging because Kubernetes has many non-secure by default settings. These difficulties create a gap between DevOps and InfoSec. Kubernetes admins are not trained security professionals and don't necessarily understand the security implications of different settings, while InfoSec teams know even less about Kubernetes and its architecture.

With so many moving parts and a broad attack surface to take into consideration, there is no simple answer to "How can we secure our Kubernetes environment?" Complete security coverage means understanding the security risks that impact a containerized environment, and specifically in a Kubernetes-based environment.



Kubernetes Threat Matrix

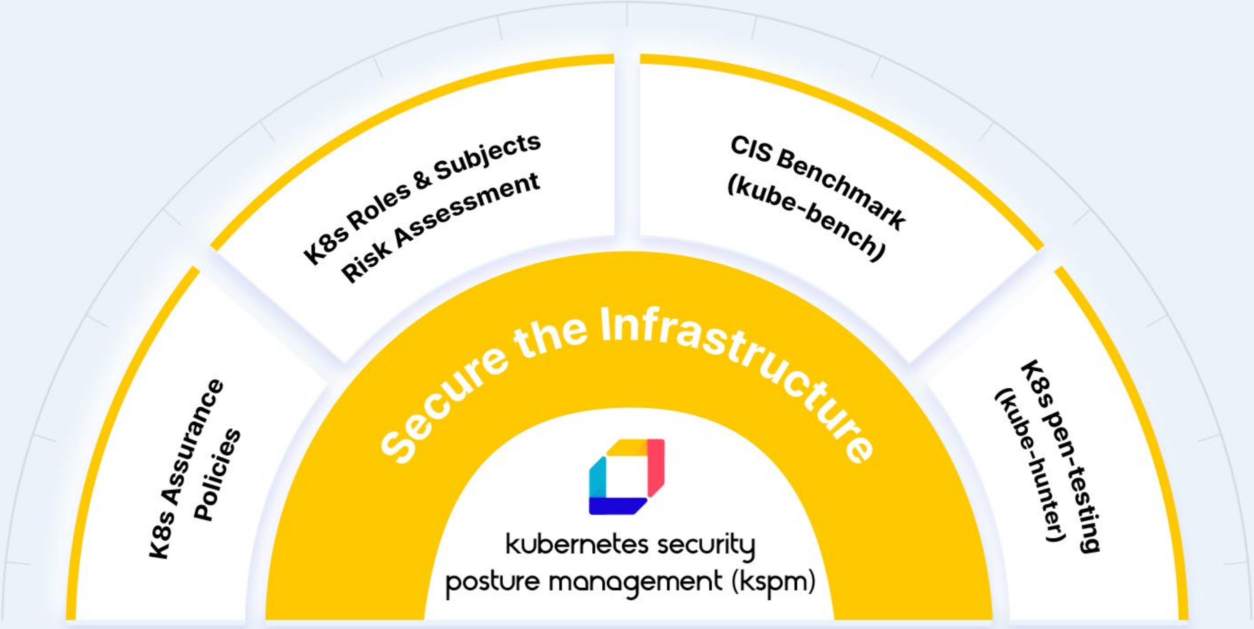
The MITRE ATT&CK® framework uses a knowledge base of known tactics and techniques used in cyberattacks to show how an attacker might breach an environment. The following table provides a guide for SecOps and InfoSec professionals to the critical elements of securing a k8s environment aligned to the MITRE ATT&CK categories.

MITRE ATT&CK framework category	Kubernetes security context	Kubernetes attack vector
Initial access	Initial access consists of techniques that are used for gaining access to the cluster. This access can be achieved directly via the cluster management layer or by gaining access to a malicious or vulnerable resource that is deployed on the cluster	• Using cloud credentials • Compromised images in the registry • Kubeconfig file • Misconfigured API or Docker Daemon • Vulnerable application • Exposed dashboard
Execution	The execution vector refers to methods used to run malicious code inside a Kubernetes cluster once the adversary gains access to the cluster.	• Shell access /command inside the container • New container • Application exploit • Remote access to a container • Exec into container
Persistence	Persistence tactics enable access to the cluster to be maintained even if the initial foothold is lost.	• Writable hostPath mount • Kubernetes CronJob • Backdoor container
Privilege escalation	The privilege escalation tactic consists of techniques that include getting access to the node from a container, gaining higher privileges in the cluster, and even getting access to the cloud resources.	• Cluster-admin binding • hostPath mount • Access cloud resources • Privileged container • Docker escape
Defense evasion	When attackers avoid detection and hide their activity.	• Connect from a proxy server • Delete Kubernetes events • Base64 encoded / encrypted malware
Credential access	Attackers steal credentials of the running application, identities, secrets stored in the cluster, or cloud credentials.	• Mount service principal • Access container service account (SA) • Application credentials in configuration files • DNS spoofing • List Kubernetes secrets • Incorrectly configured etcd DB
Discovery	The discovery tactic consists of techniques used by attackers to explore the environment to which they gained access. This exploration helps the attackers to perform lateral movement and gain access to additional resources.	• Access Kubelet API • Network mapping • Access Kubernetes dashboard • Instance Metadata API • Etcd • Docker.sock
Lateral movement	The attacker gains access to various cluster resources from given access to one container, gaining access to the underlying node from a container/pod.	• Container service account • Cluster internal networking • Applications credentials in configuration files • Writable volume mounts on the host • Access Kube-Proxy • Access tiller endpoint • Access cloud resources
Impact	When attackers ultimately have gained enough access to destroy, abuse, or disrupt the environment's normal behavior.	• Resource hijacking • Denial of service • Data destruction • Credential theft • Data exfiltration

Introducing Kubernetes Security Posture Management

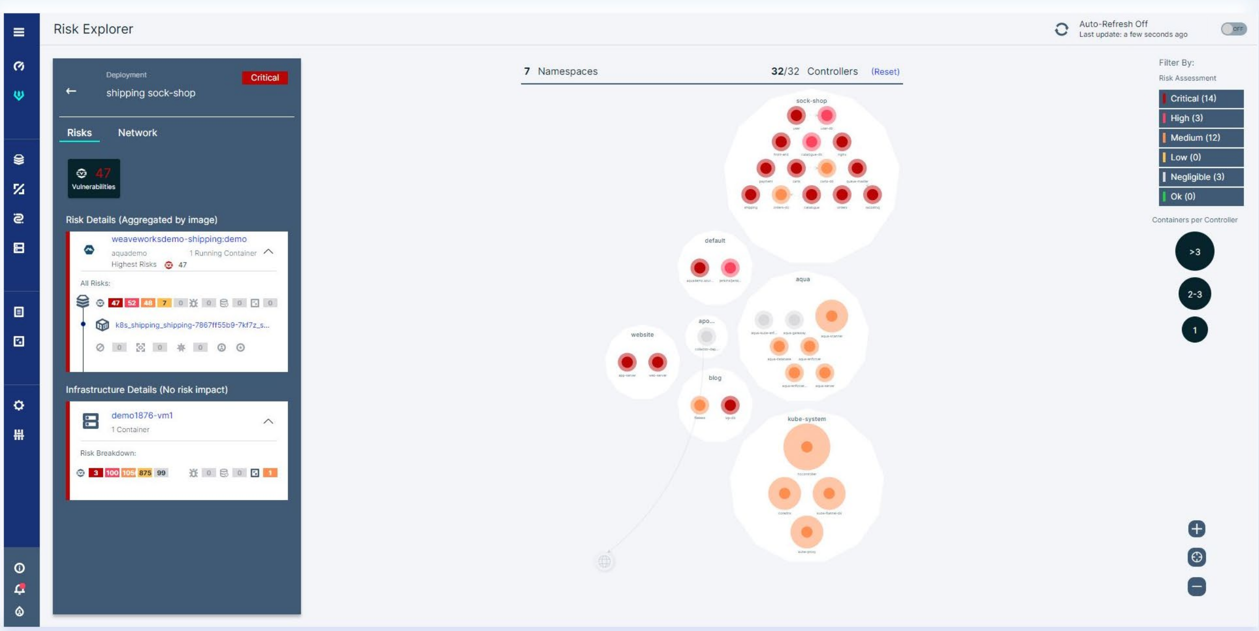
Aqua's Kubernetes Security Posture Management (KSPM) is a new, holistic approach to ensuring the ongoing security and compliance posture of your Kubernetes infrastructure. It minimizes the Kubernetes attack surface, prevents administrator errors, and protects against common potential attack vectors described by the MITRE ATT@CK framework.

Aqua KSPM leverages and extends native Kubernetes capabilities, enabling security and compliance teams to enforce policy-driven secure configuration and governance. It empowers organizations to identify and remediate risks through continuous security assessment and remediation advice, securing the essential orchestration layer of cloud native applications.


K8s Assurance Policies
K8s Roles & Subjects Risk Assessment
CIS Benchmark (kube-bench)
K8s pen-testing (kube-hunter)
Secure the Infrastructure
kubernetes security posture management (kspm)

Visualizing risks in your cluster

Risk Explorer provides DevOps admins a dynamic display of all running workloads and the security risks associated with each workload. A clear, aggregated visualization of risks in your Kubernetes aids with the analysis of those risks in running workloads. From a single view, you can review the namespaces, deployments, nodes (hosts), containers (and the images they came from), as well as network connections between and within namespaces along with their calculated risk scores.



Risk Explorer

Auto-Refresh Off
Last update: a few seconds ago

Deployment
shipping sock-shop
Critical

Risks | Network

47 Vulnerabilities

Risk Details (Aggregated by image)

weaveworksdemo-shipping:demo
aquademo — 1 Running Container
Highest Risks 47

All Risks:
47 52 48 7 0 | 0 | 0 | 0

k8s_shipping_shipping-7867ff55b9-7kf7z_s...
0 | 0 | 0

Infrastructure Details (No risk impact)

demo1876-vm1
1 Container

Risk Breakdown:
3 100 105 875 99 | 0 | 0 | 1

7 Namespaces — 32/32 Controllers (Reset)

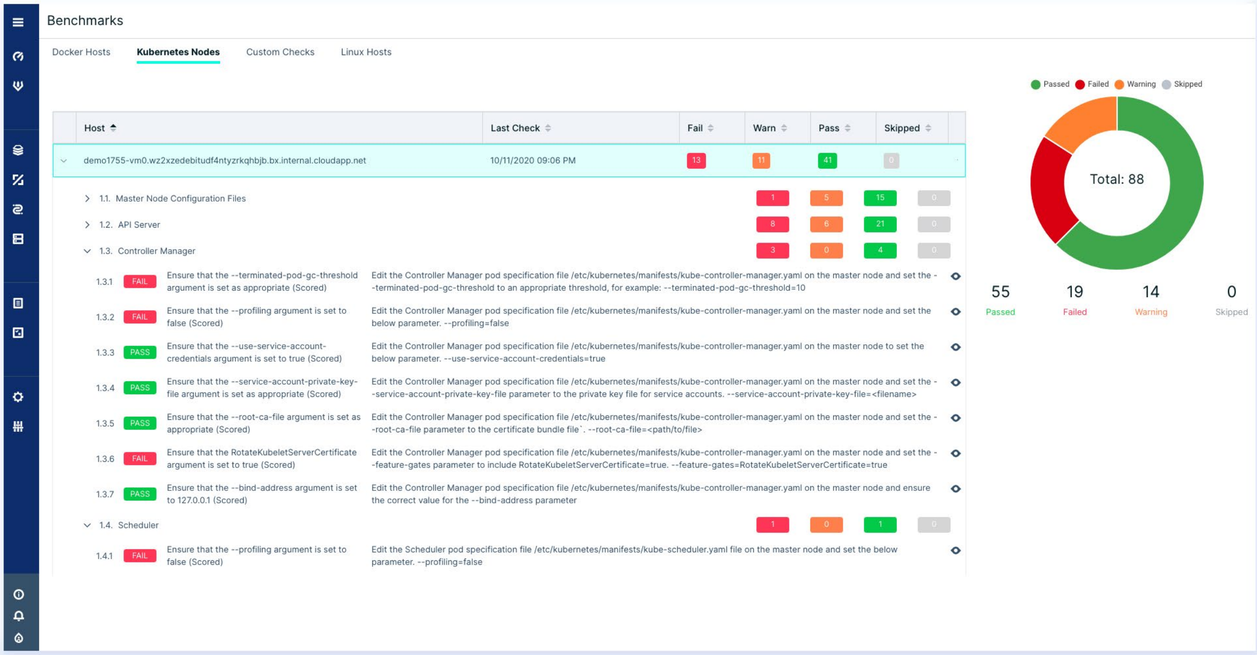
Filter By:
Risk Assessment
- Critical (14)
- High (3)
- Medium (12)
- Low (0)
- Negligible (3)
- Ok (0)

Containers per Controller
- >3
- 2-3
- 1

Hardening Your Kubernetes Cluster

Ensure that your Kubernetes' components and configurations meet security best practices, as defined in the CIS Kubernetes Benchmark. Powered by Aqua's open-source tool Kube-bench, KSPM provides security guidelines and tests for establishing a secure Kubernetes configuration posture. Quickly get up to speed on key considerations for reliability and security and adhere to best practices in critical areas, including:

- Issues with user authentication, permissions, and secure data, among other areas
- Securing data in transit and at rest
- Using least privileges

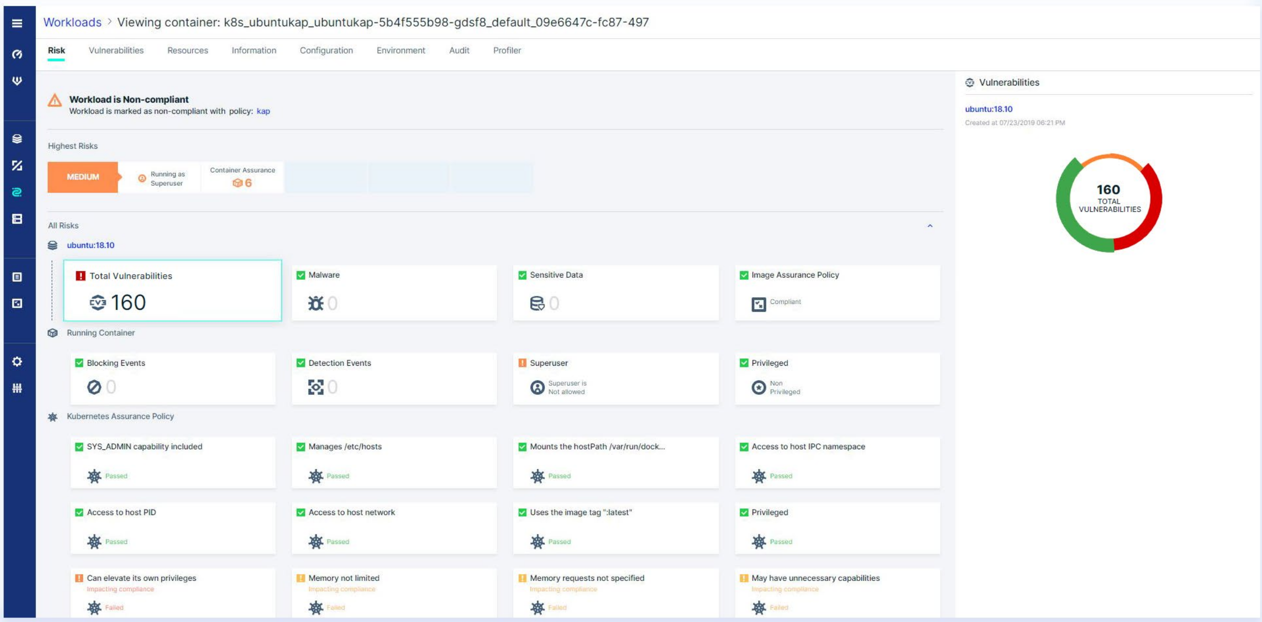


Host	Last Check	Fail	Warn	Pass	Skipped
demo1755-vm0.wz2xzedebitudf4ntyzrkqhbjb.bx.internal.cloudapp.net	10/11/2020 09:06 PM	13	11	41	0
1.1. Master Node Configuration Files		1	5	15	0
1.2. API Server		8	6	21	0
1.3. Controller Manager		3	0	4	0
1.3.1 FAIL Ensure that the --terminated-pod-gc-threshold argument is set as appropriate (Scored)	Edit the Controller Manager pod specification file /etc/kubernetes/manifests/kube-controller-manager.yaml on the master node and set the --terminated-pod-gc-threshold to an appropriate threshold, for example: --terminated-pod-gc-threshold=10				
1.3.2 FAIL Ensure that the --profiling argument is set to false (Scored)	Edit the Controller Manager pod specification file /etc/kubernetes/manifests/kube-controller-manager.yaml on the master node and set the below parameter. --profiling=false				
1.3.3 PASS Ensure that the --use-service-account-credentials argument is set to true (Scored)	Edit the Controller Manager pod specification file /etc/kubernetes/manifests/kube-controller-manager.yaml on the master node to set the below parameter. --use-service-account-credentials=true				
1.3.4 PASS Ensure that the --service-account-private-key-file argument is set as appropriate (Scored)	Edit the Controller Manager pod specification file /etc/kubernetes/manifests/kube-controller-manager.yaml on the master node and set the --service-account-private-key-file parameter to the private key file for service accounts. --service-account-private-key-file=<filename>				
1.3.5 PASS Ensure that the --root-ca-file argument is set as appropriate (Scored)	Edit the Controller Manager pod specification file /etc/kubernetes/manifests/kube-controller-manager.yaml on the master node and set the --root-ca-file parameter to the certificate bundle file`. --root-ca-file=<path/to/file>				
1.3.6 FAIL Ensure that the RotateKubeletServerCertificate argument is set to true (Scored)	Edit the Controller Manager pod specification file /etc/kubernetes/manifests/kube-controller-manager.yaml on the master node and set the --feature-gates parameter to include RotateKubeletServerCertificate=true. --feature-gates=RotateKubeletServerCertificate=true				
1.3.7 PASS Ensure that the --bind-address argument is set to 127.0.0.1 (Scored)	Edit the Controller Manager pod specification file /etc/kubernetes/manifests/kube-controller-manager.yaml on the master node and ensure the correct value for the --bind-address parameter				
1.4. Scheduler		1	0	1	0
1.4.1 FAIL Ensure that the --profiling argument is set to false (Scored)	Edit the Scheduler pod specification file /etc/kubernetes/manifests/kube-scheduler.yaml file on the master node and set the below parameter. --profiling=false				

Shifting-left workload security

While scanning images in CI/CD and registries has become a widely adopted practice, it only addresses vulnerabilities and other security issues inside container images. However, the risks of improperly configured workloads running in Kubernetes are often neglected, and organizations lack automated, policy-driven security gates that enforce best practices across development and operations, before workloads are deployed.

Aqua Kubernetes Assurance Policies allow you to evaluate specific conditions related to your workloads and check for potential unsafe security configurations, whether in your cluster, node or pod. Compatible with Open Policy Agent (OPA) and using Rego expressions, it provides out-of-the-box rules and allows you to add custom Rego rules to comply with your security requirements. Aqua Kubernetes Assurance Policies introduce a new flow that enables only allowing workloads that comply with both their image assurance policy, addressing issues such as vulnerabilities, malware, and container root access, as well as with K8s configuration requirements, such as Pod resource use or networking privileges. Together these policies create a robust preventive mechanism for allow-listing authorized workloads.



Workloads > Viewing container: k8s_ubuntukap_ubuntukap-5b4f555b98-gdsf8_default_09e6647c-fc87-497

Workload is Non-compliant
Workload is marked as non-compliant with policy: kap

Highest Risks
MEDIUM | Running as Superuser | Container Assurance 6

All Risks
ubuntu:18.10

- Total Vulnerabilities: 160
- Malware: 0
- Sensitive Data: 0
- Image Assurance Policy: Compliant

Running Container

- Blocking Events: 0
- Detection Events: 0
- Superuser: Superuser is Not allowed
- Privileged: Non Privileged

Kubernetes Assurance Policy

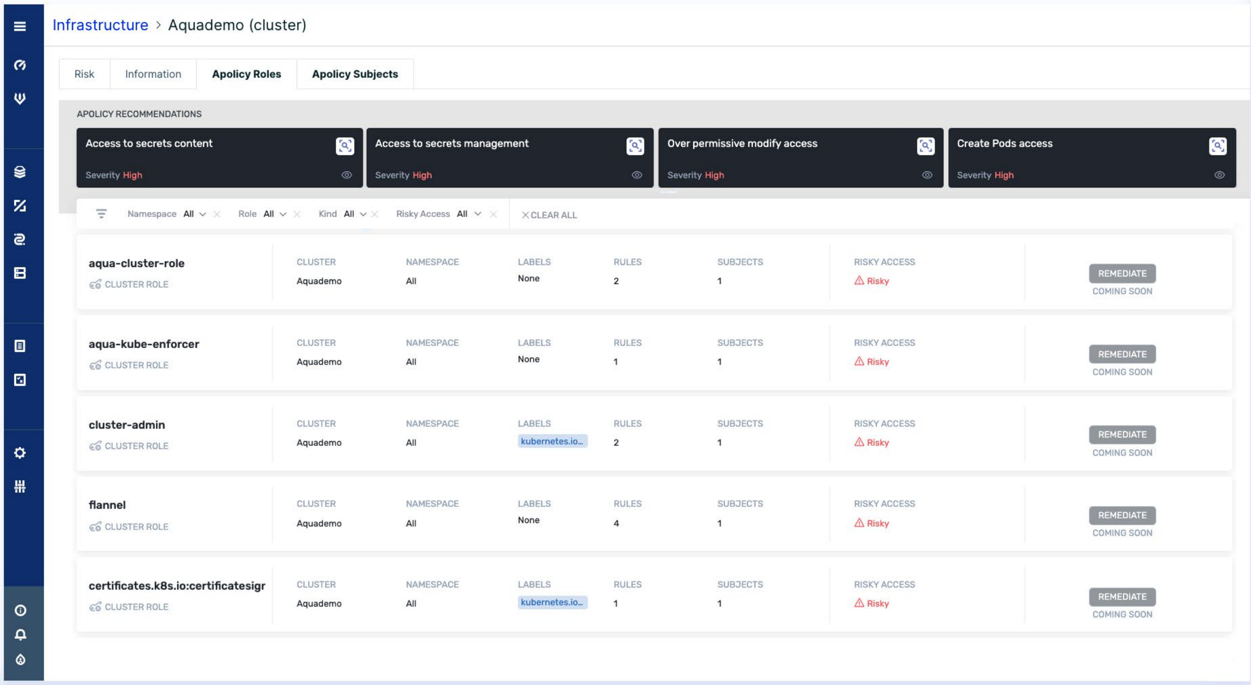
- SYS_ADMIN capability included: Passed
- Manages /etc/hosts: Passed
- Mounts the hostPath /var/run/dock...: Passed
- Access to host IPC namespace: Passed
- Access to host PID: Passed
- Access to host network: Passed
- Uses the image tag ":latest": Passed
- Privileged: Passed
- Can elevate its own privileges: Impacting compliance, Failed
- Memory not limited: Impacting compliance, Failed
- Memory requests not specified: Impacting compliance, Failed
- May have unnecessary capabilities: Impacting compliance, Failed

Vulnerabilities
ubuntu:18.10
Created at 07/23/2019 06:21 PM
160 TOTAL VULNERABILITIES

Reducing the Risk of Permissive Roles & Subjects

One of the key risk areas for Kubernetes environments is the overly permissive defaults that users and K8s subjects, such as service accounts, may have. Wrangling these permissions and enforcing the least privileges manually or using dozens, if not hundreds of YAML files is time-consuming, prone to human error, and ultimately introducing significant risk. Aqua Kubernetes Roles & Subjects Assessment help DevOps teams to ensure least-privilege access in their K8s environment while maintaining the right privileges for every user. Aqua Kubernetes Roles & Subjects Assessment reduces the tension points between security and operations, automate policy enforcement at scale, enable governance, and avoid compliance issues.

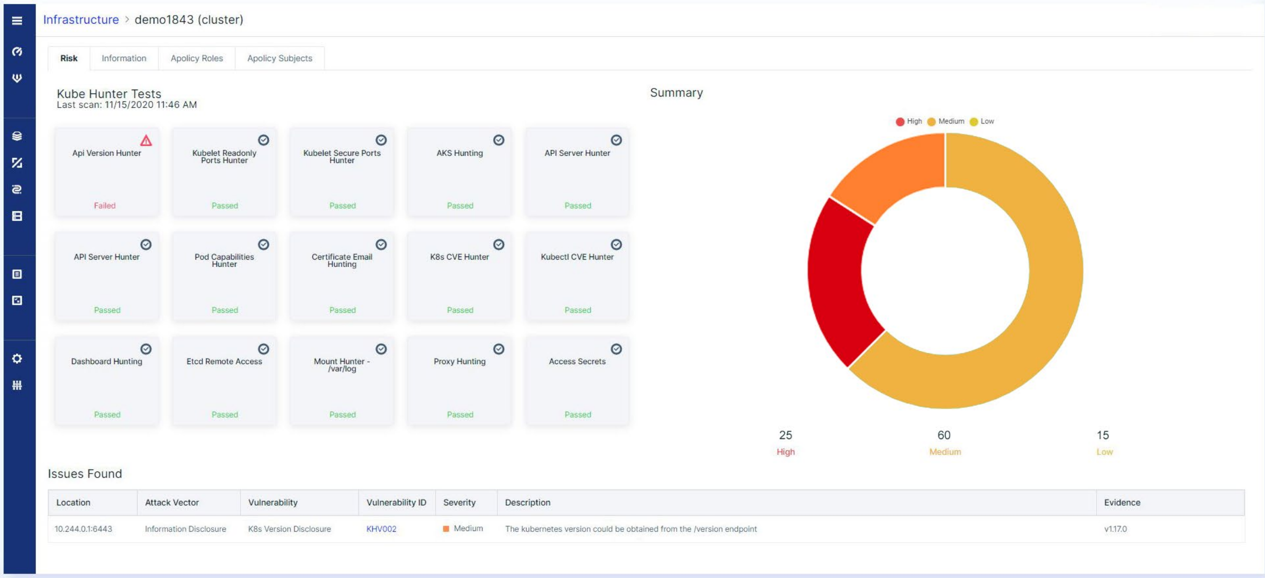
- Gain deep visibility and insights while providing DevOps teams with control over their environment
- Assess workload exposure and prioritize risk for actions
- Address least privilege security gaps and make remediation recommendations based on best practices
- View remediation advice to manage risk



	CLUSTER	NAMESPACE	LABELS	RULES	SUBJECTS	RISKY ACCESS	
aqua-cluster-role CLUSTER ROLE	Aquademo	All	None	2	1	⚠ Risky	REMEDIATE COMING SOON
aqua-kube-enforcer CLUSTER ROLE	Aquademo	All	None	1	1	⚠ Risky	REMEDIATE COMING SOON
cluster-admin CLUSTER ROLE	Aquademo	All	kubernetes.io...	2	1	⚠ Risky	REMEDIATE COMING SOON
flannel CLUSTER ROLE	Aquademo	All	None	4	1	⚠ Risky	REMEDIATE COMING SOON
certificates.k8s.io:certificatesigr CLUSTER ROLE	Aquademo	All	kubernetes.io...	1	1	⚠ Risky	REMEDIATE COMING SOON

Pen-Testing Kubernetes Again Known Attack Vectors

DevOps and SecOps teams need to identify weaknesses in their K8s clusters that go beyond simple configuration tests, finding those that can be exploited and remediating them before adversaries take advantage of them. Based on Aqua's open source Kube Hunter, this feature runs automated penetration tests on your Kubernetes clusters, based on dozens of constantly updated known attack vectors. Kubernetes admins can quickly eliminate blind spots and protect their K8s clusters against known exploits and attacks.



Infrastructure > demo1843 (cluster)

Risk | Information | Apolicy Roles | Apolicy Subjects

Kube Hunter Tests
Last scan: 11/15/2020 11:46 AM

Api Version Hunter Failed	Kubelet Readonly Ports Hunter Passed	Kubelet Secure Ports Hunter Passed	AKS Hunting Passed	API Server Hunter Passed
API Server Hunter Passed	Pod Capabilities Hunter Passed	Certificate Email Hunting Passed	K8s CVE Hunter Passed	Kubectl CVE Hunter Passed
Dashboard Hunting Passed	Etcd Remote Access Passed	Mount Hunter - /var/log Passed	Proxy Hunting Passed	Access Secrets Passed

Summary

Issues Found

Location	Attack Vector	Vulnerability	Vulnerability ID	Severity	Description	Evidence
10.244.0.1:6443	Information Disclosure	K8s Version Disclosure	KHV002	■ Medium	The kubernetes version could be obtained from the /version endpoint	v1.17.0

Continuously Protect Kubernetes Clusters

The Aqua platform provides full visibility and security automation across the entire application lifecycle, using a modern, zero-touch approach to detect and prevent threats while simplifying regulatory compliance. Aqua works across all clouds and platforms, securing workloads that run across containers, VMs, and serverless functions.

As part of that, Aqua KSPM provides a holistic view of the security posture of your Kubernetes infrastructure for accurate reporting and remediation. As security concerns and misconfigurations remain the leading concerns regarding the use of containers and Kubernetes, KSPM helps organizations form and implement a consistent and automated an in-depth defense. Organizations can secure their Kubernetes cluster configurations, understand, and reduce the k8s attack surface, apply least-privilege access best practices, avoid human errors, identify and remediate risks, and prove compliance with regulations.

Cluster-wide visibility	Risk assessment & compliance	Automated policy enforcement
Gain a clear view of the security posture of the Kubernetes environment across its entire architecture: • host components • internal configuration • management API • workload pods Increase visibility into your Kubernetes environment and expose any policy violation. Raise awareness and visibility for security issues in Kubernetes environments	Assess workload exposure and prioritize risk for actions, preventing issues before they arise Evaluate data risk and detect growing security risks in account permissions Align with compliance requirements of common standards and best practices, such as CIS Benchmarks, PCI, NIST 800-53, or HIPAA	Leverage Rego and OPA capabilities and make it easy for DevOps teams to implement policies in Aqua Reduce remediation time for any issue that might hide in your cluster Alert on potential security misconfigurations to SecOps admins who can appease the problem immediately

Go Cloud Native with the Experts!

Get a Demo >

aquasec.com

contact@aquasec.com

@Aqua Security

@AquaSecTeam

in/Aqua Security

@AquaSecTeam