portshift

CLOUD NATIVE COMPUTING FOUNDATION
SILVER MEMBER

Stay On Top Of Ongoing Kubernetes Security Hygiene

Zohar Kaufman
Ariel Shuper
July 2020

Discussion Points

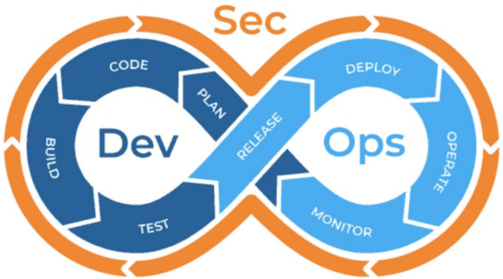
- ➔ Vulnerabilities Scan: A new approach
- ➔ Advanced Runtime Risks Detection: CD pipelines
- ➔ Pods Deployments: The Good, The Complex and The Intuitive way
- ➔ Network policies: There’s a Life Outside the Cluster
- ➔ Kubernetes Master node Security: Really?
- ➔ Wrap Up + Shareables
- ➔ Q&A

Vulnerabilities Scan:

A new approach

Classical Image Scan

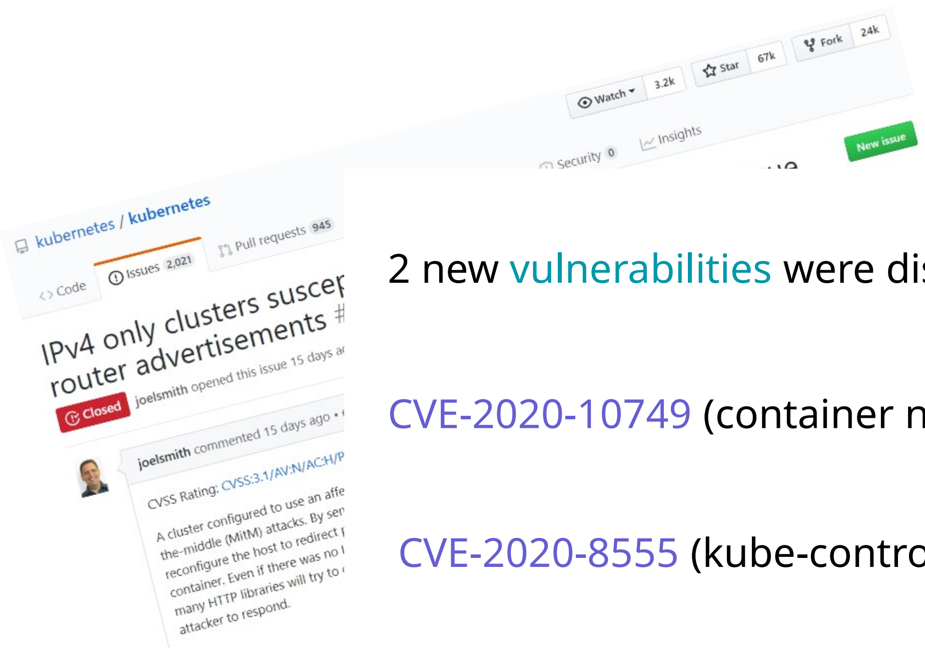
- Scan in registry or with a pipeline job
- Risks are detected in advanced
- Developers can fix/replace quicker and more efficiently
- DevSecOps do not need to communicate to developers that they shouldn’t use certain images
- Preventing vulnerable pods from being deployed



The Challenges of the Classical Approach

1. Require preliminary integrations
2. External repositories & Sidecars
3. Inefficient process
4. New Vulnerabilities can be found after deployment
5. The Kubernetes Infrastructure may be vulnerable too

A New Approach is Required



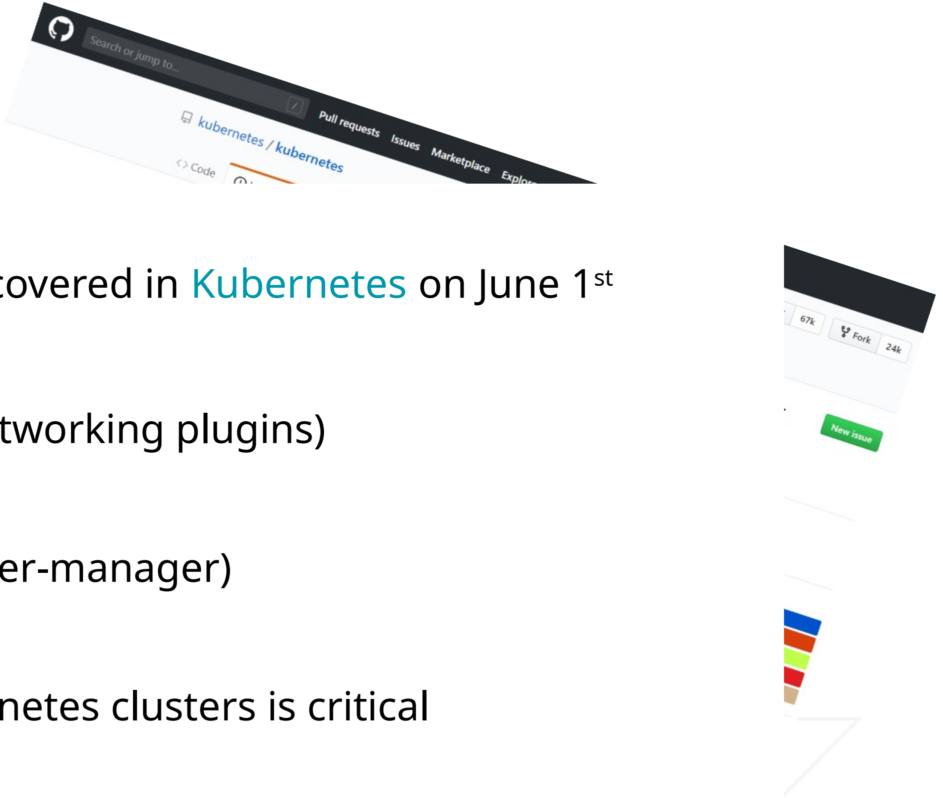
IPv4 only clusters suscep
router advertisements #

joelsmith opened this issue 15 days a

joelsmith commented 15 days ago •

CVSS Rating: CVSS:3.1/AV:N/AC:H/P

A cluster configured to use an affe
the-middle (MitM) attacks. By sen
reconfigure the host to redirect
container. Even if there was no
many HTTP libraries will try to
attacker to respond.



2 new vulnerabilities were discovered in Kubernetes on June 1st

CVE-2020-10749 (container networking plugins)

CVE-2020-8555 (kube-controller-manager)

Constant monitoring of Kubernetes clusters is critical

portshift

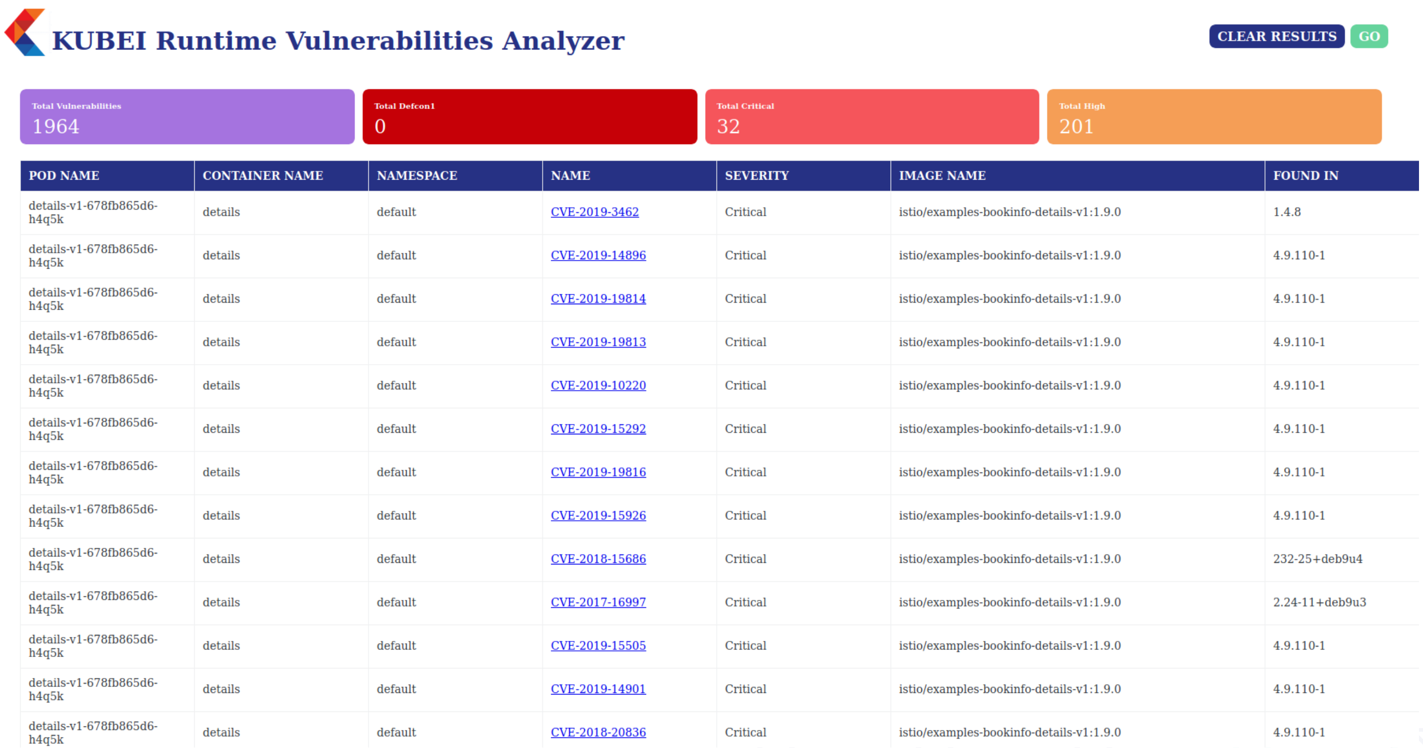
Introducing Kubei, an Open Source Runtime Scanner

- Scan all runtime images, whether coming from CI/CD or not
- Detect malicious pods and provide an extra layer of security to your cluster
- Scan public images you use that are not hosted in your registry
- Gain visibility into Kubernetes elements which suffer from occasional vulnerabilities
- Get accurate, real-time status of your cluster’s health

How Kubei Works

- No need for registry or CI/CD integration
- Kubei runs inside the cluster and provides real-time assessment of deployed containers’ vulnerabilities
- Distributes the load
- Generates a report that will show the exact location of the pods, their risks, and the relevant packages to be patched

Demo



KUBEI Runtime Vulnerabilities Analyzer

Total Vulnerabilities 1964

Total Defcon1 0

Total Critical 32

Total High 201

POD NAME	CONTAINER NAME	NAMESPACE	NAME	SEVERITY	IMAGE NAME	FOUND IN
details-v1-678fb865d6-h4q5k	details	default	CVE-2019-3462	Critical	istio/examples-bookinfo-details-v1:1.9.0	1.4.8
details-v1-678fb865d6-h4q5k	details	default	CVE-2019-14896	Critical	istio/examples-bookinfo-details-v1:1.9.0	4.9.110-1
details-v1-678fb865d6-h4q5k	details	default	CVE-2019-19814	Critical	istio/examples-bookinfo-details-v1:1.9.0	4.9.110-1
details-v1-678fb865d6-h4q5k	details	default	CVE-2019-19813	Critical	istio/examples-bookinfo-details-v1:1.9.0	4.9.110-1
details-v1-678fb865d6-h4q5k	details	default	CVE-2019-10220	Critical	istio/examples-bookinfo-details-v1:1.9.0	4.9.110-1
details-v1-678fb865d6-h4q5k	details	default	CVE-2019-15292	Critical	istio/examples-bookinfo-details-v1:1.9.0	4.9.110-1
details-v1-678fb865d6-h4q5k	details	default	CVE-2019-19816	Critical	istio/examples-bookinfo-details-v1:1.9.0	4.9.110-1
details-v1-678fb865d6-h4q5k	details	default	CVE-2019-15926	Critical	istio/examples-bookinfo-details-v1:1.9.0	4.9.110-1
details-v1-678fb865d6-h4q5k	details	default	CVE-2018-15686	Critical	istio/examples-bookinfo-details-v1:1.9.0	232-25+deb9u4
details-v1-678fb865d6-h4q5k	details	default	CVE-2017-16997	Critical	istio/examples-bookinfo-details-v1:1.9.0	2.24-11+deb9u3
details-v1-678fb865d6-h4q5k	details	default	CVE-2019-15505	Critical	istio/examples-bookinfo-details-v1:1.9.0	4.9.110-1
details-v1-678fb865d6-h4q5k	details	default	CVE-2019-14901	Critical	istio/examples-bookinfo-details-v1:1.9.0	4.9.110-1
details-v1-678fb865d6-h4q5k	details	default	CVE-2018-20836	Critical	istio/examples-bookinfo-details-v1:1.9.0	4.9.110-1

portshift

Advanced Risk Detection: CD Pipelines

CD: Advanced detection of potential risks

- Add a new security check step Integrated in the CD pipeline
- Assess the risk potential prior to its deployment
    - Pod security context: Are we running as a root?
    - RBAC permissions: Do you really need to update configmap?
    - Secrets: Did you leave them exposed?

HELM

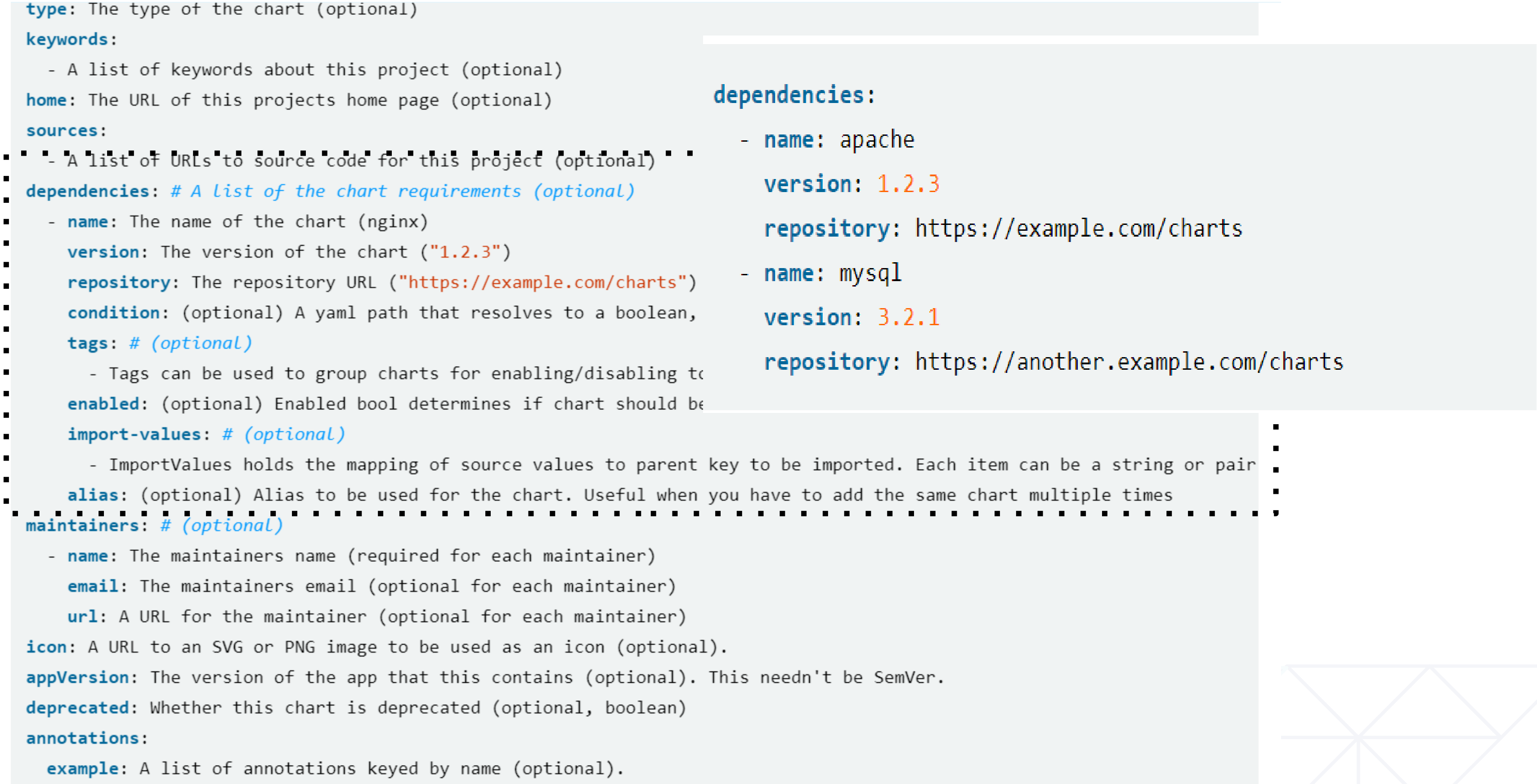
Bamboo

Travis CI

TeamCity

Jenkins Azure Pipelines argo Spinnaker circleci TEKTON GitLab

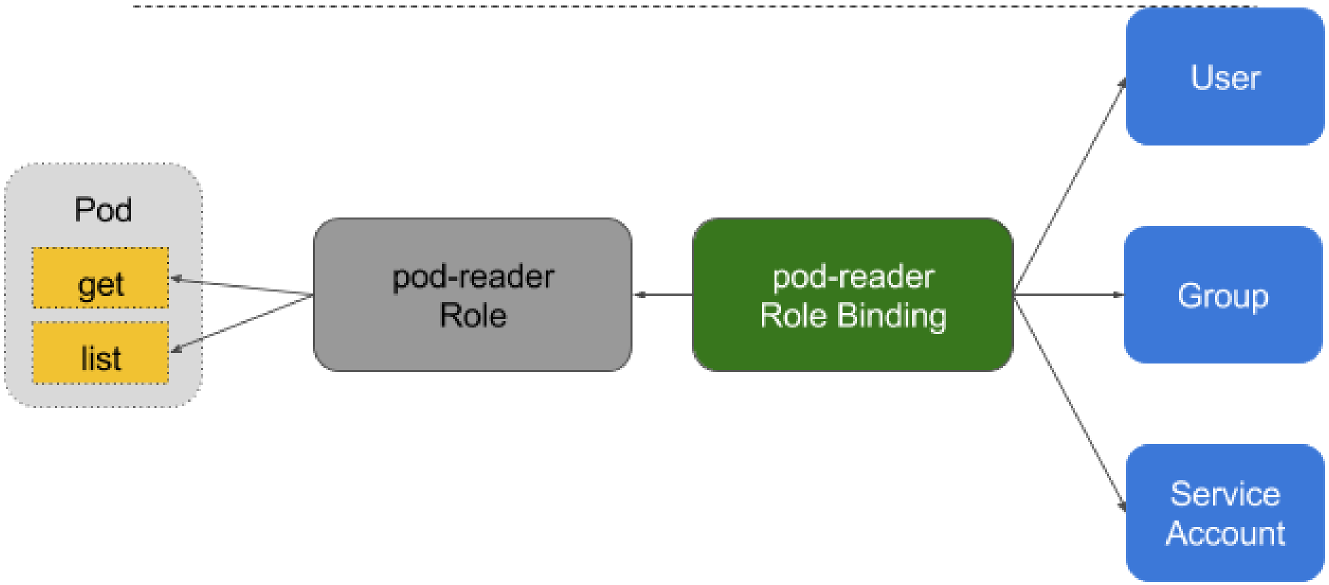
Example of Advanced Risk Detection



```
type: The type of the chart (optional)
keywords:
  - A list of keywords about this project (optional)
home: The URL of this projects home page (optional)
sources:
  - A list of URLs to source code for this project (optional)
dependencies: # A list of the chart requirements (optional)
  - name: The name of the chart (nginx)
    version: The version of the chart ("1.2.3")
    repository: The repository URL ("https://example.com/charts")
    condition: (optional) A yaml path that resolves to a boolean,
    tags: # (optional)
      - Tags can be used to group charts for enabling/disabling tc
    enabled: (optional) Enabled bool determines if chart should be
    import-values: # (optional)
      - ImportValues holds the mapping of source values to parent key to be imported. Each item can be a string or pair
    alias: (optional) Alias to be used for the chart. Useful when you have to add the same chart multiple times
maintainers: # (optional)
  - name: The maintainers name (required for each maintainer)
    email: The maintainers email (optional for each maintainer)
    url: A URL for the maintainer (optional for each maintainer)
icon: A URL to an SVG or PNG image to be used as an icon (optional).
appVersion: The version of the app that this contains (optional). This needn't be SemVer.
deprecated: Whether this chart is deprecated (optional, boolean)
annotations:
  example: A list of annotations keyed by name (optional).
```

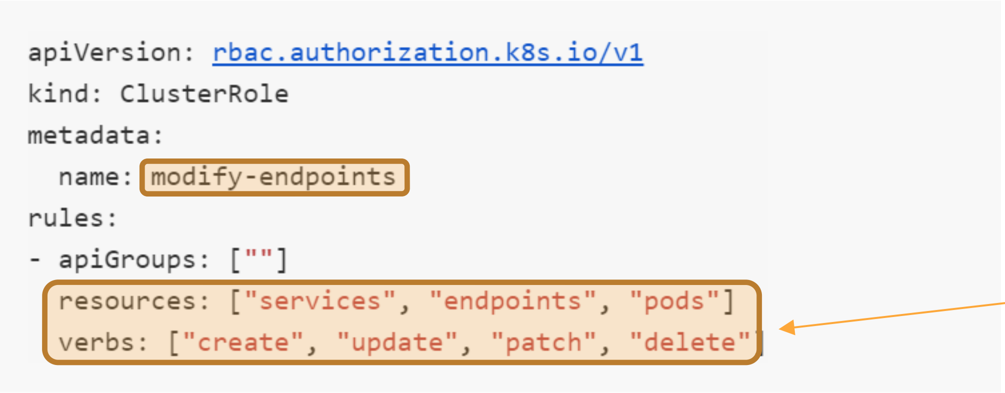
```
dependencies:
  - name: apache
    version: 1.2.3
    repository: https://example.com/charts
  - name: mysql
    version: 3.2.1
    repository: https://another.example.com/charts
```

Role Based Access Control in Kubernetes

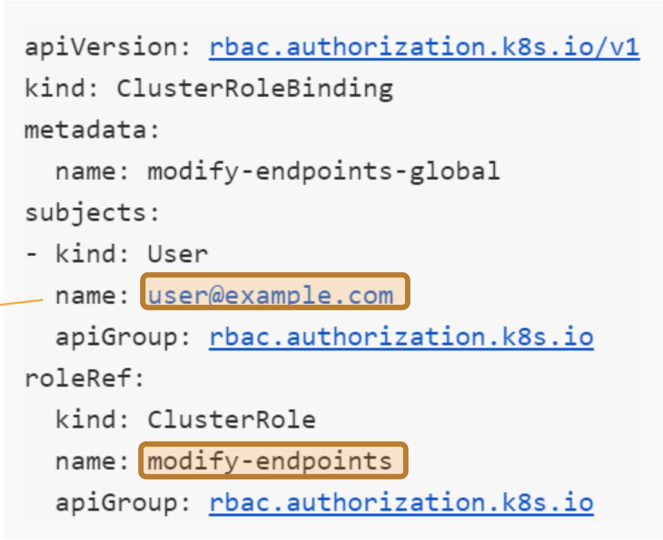


RBAC: What Can Go Wrong?

- Allowing user@example.com to modify services, endpoints and pods in the whole cluster



```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: modify-endpoints
rules:
- apiGroups: [""]
  resources: ["services", "endpoints", "pods"]
  verbs: ["create", "update", "patch", "delete"]
```



```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: modify-endpoints-global
subjects:
- kind: User
  name: user@example.com
  apiGroup: rbac.authorization.k8s.io
roleRef:
  kind: ClusterRole
  name: modify-endpoints
  apiGroup: rbac.authorization.k8s.io
```

RBAC: Let’s Play it Secure

- Review RBAC in all clusters according to their risk classification
- Review all Users, Groups, ServiceAccount and their roles
- Create a Secure Policy based on:
    - Recommended profiles
    - Runtime events
    - Manual configuration
- Get runtime events of violations of custom and recommended rules

Pods Deployments:

The Good, The Complex and

The Intuitive way

Containers Attack Anatomy

Upon vulnerability exploitation, attacker will check what permissions/actions can be made on what resources

↓

Attackers will move from the breached container to the host in-order to get exposure to more containers sharing the same host

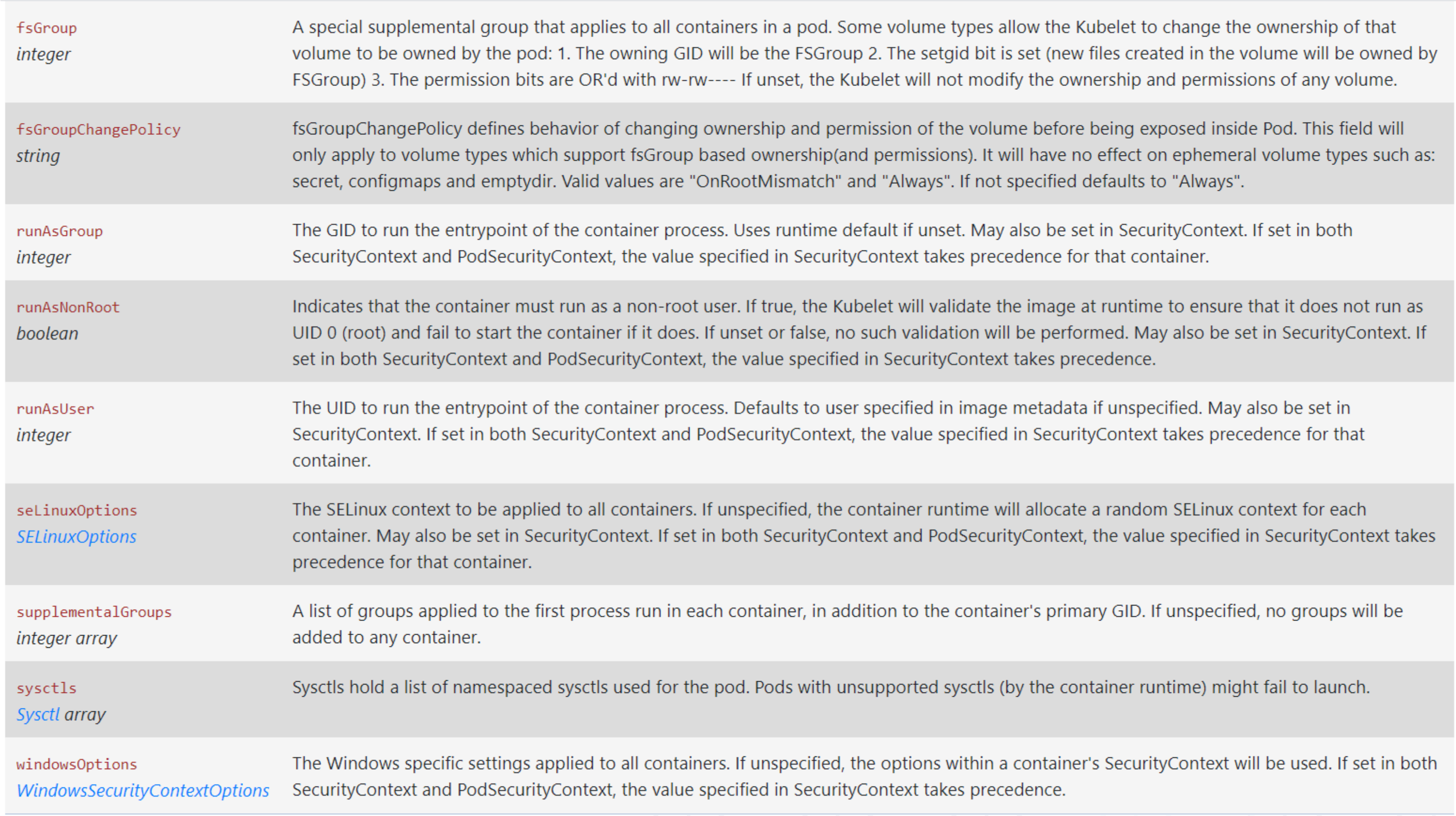
↓

Attacker will try to move to additional nodes or cluster resources to maximize the exhilaration potential

Kubernetes Pod Security Context

- ⇒ Kubernetes defines a comprehensive pod security context
- ⇒ The pod security context enable users to set up the needed configurations which will minimize the blast radius of attacks
- ⇒ Pods will be deployed with the predefined security context
- ⇒ Let’s examine the security context

Kubernetes Pod Security Context: Deep View



`fsGroup` *integer*	A special supplemental group that applies to all containers in a pod. Some volume types allow the Kubelet to change the ownership of that volume to be owned by the pod: 1. The owning GID will be the FSGroup 2. The setgid bit is set (new files created in the volume will be owned by FSGroup) 3. The permission bits are OR'd with rw-rw---- If unset, the Kubelet will not modify the ownership and permissions of any volume.
`fsGroupChangePolicy` *string*	fsGroupChangePolicy defines behavior of changing ownership and permission of the volume before being exposed inside Pod. This field will only apply to volume types which support fsGroup based ownership(and permissions). It will have no effect on ephemeral volume types such as: secret, configmaps and emptydir. Valid values are "OnRootMismatch" and "Always". If not specified defaults to "Always".
`runAsGroup` *integer*	The GID to run the entrypoint of the container process. Uses runtime default if unset. May also be set in SecurityContext. If set in both SecurityContext and PodSecurityContext, the value specified in SecurityContext takes precedence for that container.
`runAsNonRoot` *boolean*	Indicates that the container must run as a non-root user. If true, the Kubelet will validate the image at runtime to ensure that it does not run as UID 0 (root) and fail to start the container if it does. If unset or false, no such validation will be performed. May also be set in SecurityContext. If set in both SecurityContext and PodSecurityContext, the value specified in SecurityContext takes precedence.
`runAsUser` *integer*	The UID to run the entrypoint of the container process. Defaults to user specified in image metadata if unspecified. May also be set in SecurityContext. If set in both SecurityContext and PodSecurityContext, the value specified in SecurityContext takes precedence for that container.
`seLinuxOptions` *SELinuxOptions*	The SELinux context to be applied to all containers. If unspecified, the container runtime will allocate a random SELinux context for each container. May also be set in SecurityContext. If set in both SecurityContext and PodSecurityContext, the value specified in SecurityContext takes precedence for that container.
`supplementalGroups` *integer array*	A list of groups applied to the first process run in each container, in addition to the container's primary GID. If unspecified, no groups will be added to any container.
`sysctls` *Sysctl array*	Sysctls hold a list of namespaced sysctls used for the pod. Pods with unsupported sysctls (by the container runtime) might fail to launch.
`windowsOptions` *WindowsSecurityContextOptions*	The Windows specific settings applied to all containers. If unspecified, the options within a container's SecurityContext will be used. If set in both SecurityContext and PodSecurityContext, the value specified in SecurityContext takes precedence.

Kubernetes Pod Security Policy to the rescue

⇒ Kubernetes Pod Security Policy is a cluster-level resource that controls security sensitive aspects of the pod specification

⇒ The PodSecurityPolicy objects define a set of conditions that a pod must run with in order to be accepted as well as defaults for the related field

Network Policies: The Life

Outside the Cluster

Network Policies: Out of Cluster Challenges

- There are few common exceptions to classical network policies rules
- Popular one is creating a security-based network policies for resources outside the cluster:

  Which pods can access them
- Typical use cases are On-Prem Database or Cloud services (Storage)
- Another popular example is communication between clusters

The Life Outside the Cluster: Non Containerized

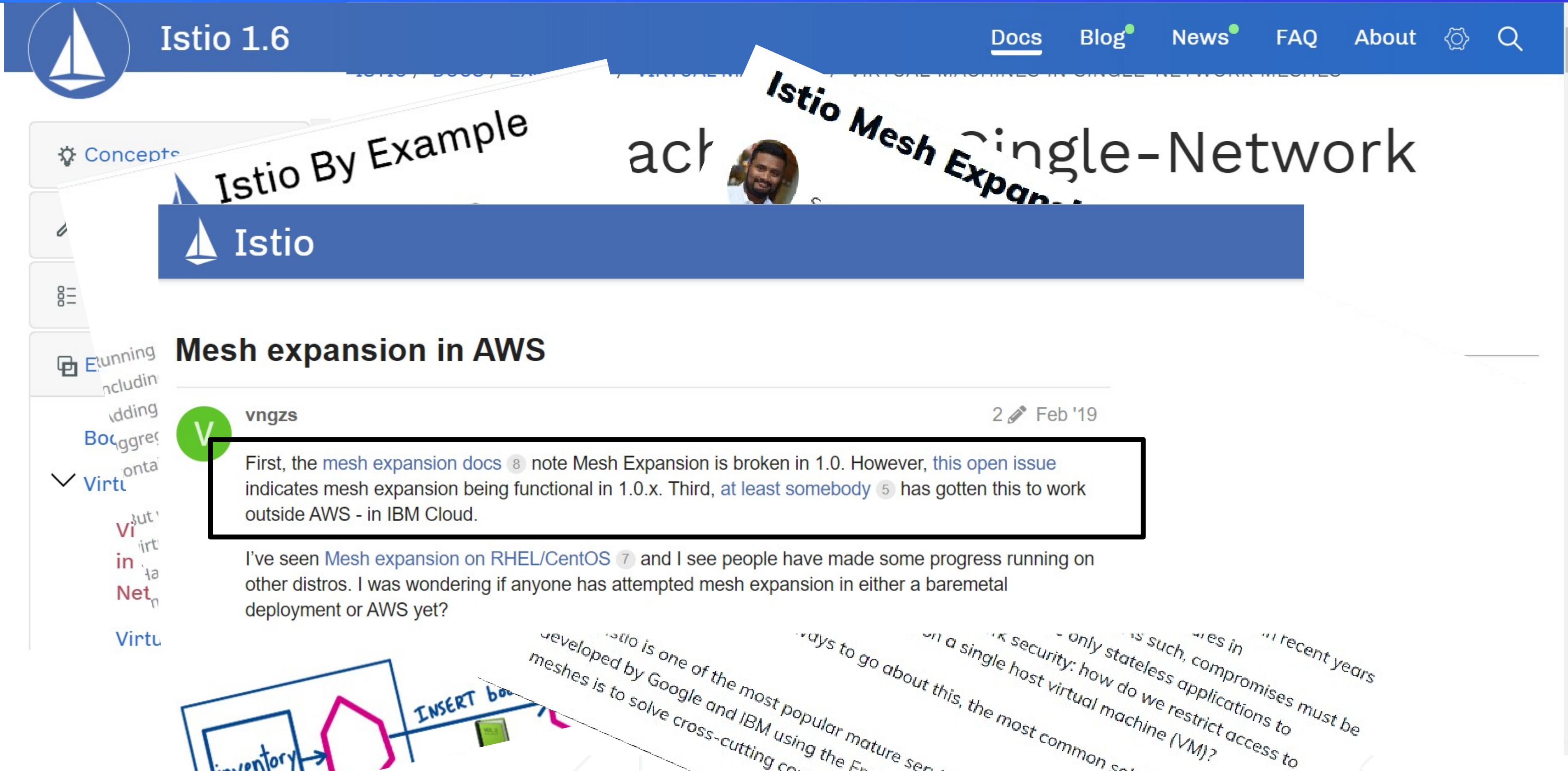
- ⇒ Service Mesh Expansion allow users to extent the Istio/service mesh to non-kubernetes elements (available since Istio 0.2)
- ⇒ The motivation for mesh-expansion is to allow containerized application to communicate with elements which aren’t containerized/part of the Kubernetes clusters
- ⇒ Many applications require to access “on-prem”/legacy resources or cloud services (Cloud-Storage, managed DBs service) which aren’t containerized

What’s Next... Service-Mesh is Already Here

- ⇒ Service mesh creates discovery and routable path to the non-Kubernetes element
- ⇒ Service mesh also provide consistent performance metrics
- ⇒ Most important: Service mesh allow users to create secured access control and encrypted channel to resources that typically contain sensitive information

Service Mesh: The Hard Way

CONFIDENTIAL


Istio 1.6
Docs
Blog
News
FAQ
About
Concepts
Istio By Example
Istio Mesh Expan
ingle-Network
Istio
INSERT
inventory
istio is one of the most popular mature ser
developed by Google and IBM using the En
meshes is to solve cross-cutting co
ways to go about this, the most common sol
on a single host virtual machine (VM)?
k security: how do we restrict access to
only stateless applications to
s such, compromises must be
in recent years

Mesh expansion in AWS

vngzs 2 Feb '19

First, the mesh expansion docs 8 note Mesh Expansion is broken in 1.0. However, this open issue indicates mesh expansion being functional in 1.0.x. Third, at least somebody 5 has gotten this to work outside AWS - in IBM Cloud.

I've seen Mesh expansion on RHEL/CentOS 7 and I see people have made some progress running on other distros. I was wondering if anyone has attempted mesh expansion in either a baremetal deployment or AWS yet?

Service Mesh: The Intuitive Way

- To simplify the process we wrote a script that automates all the steps (updating it for each version)
- Few tips:
  - Create a “virtual pod” within an existing namespace in the cluster (for mTLS certificate), to get mTLS certificate which will be used by the VM
  - Create installer that copies the certificate and additional settings to the VM
  - The installer installs Istio-expansion and the Envoy on the VM using the copied attributes
  - Synch all services to be accessible from the cluster and from the VM to allow communication

Cluster Security: K8s Master

Protection

Kubernetes API and RBAC: Why Do We Need It?

K8S Server API

Resource access via HTTPS API for users and workloads
(pods, secrets, configmaps, nodes, services, etc.)

"kubectl edit pod -n my-namespace my-pod"

PATCH https://34.71.145.93
/api/v1/namespaces/my-namespace
/pods/my-pod

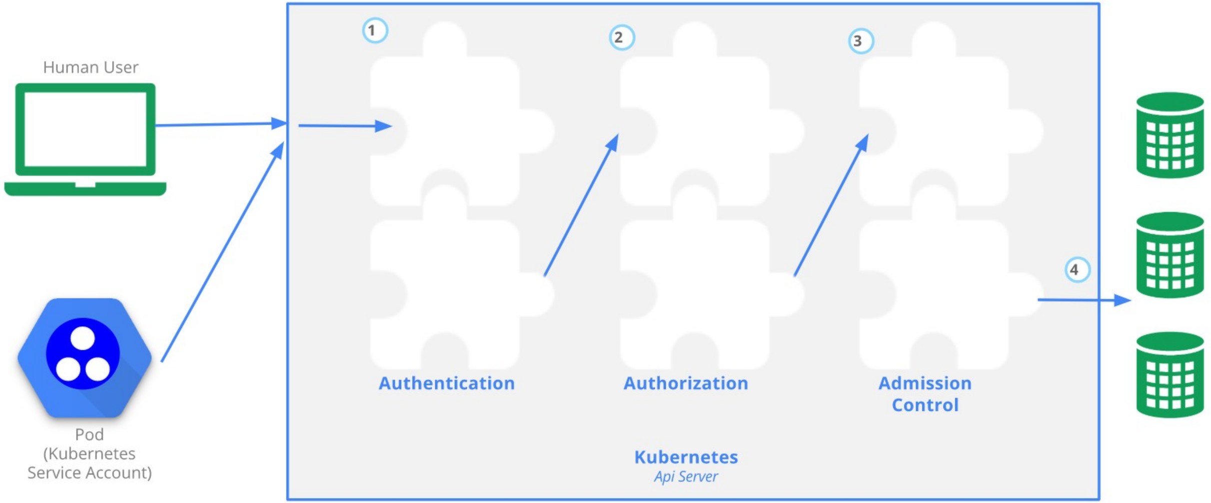
RBAC (Role based access control)

Who can do **what, where** and in which **action**

Who *- user/group/serviceAccount*
What *– API group & resource*
Where *- namespace/cluster*
Action *- create, get, update, etc.*

portshift

API Server architecture flow



etcd
Controls
Scheduler

The Ideal Protection Scheme

- Runtime audit to all APIs invoked towards API server with associated risk
- Block commands that create risks or violate security best-practices
- Detect abnormal scenarios

Demo

portshift

Summary

1. Accurately assess the runtime environment - It is critical for your Kubernetes security health
2. Kubernetes security scope is broader than image-scanning, additional attributes need to be taken into considerations to maintain a high level of security
3. Network policies, are not just communication enforcement methods, they can be used wisely to govern the internal and external access potential exposure

Wrap Up + Shareables

https://github.com/Portshift/Kubei

https://portshift.io/runtime-kubernetes-scanning-kubei/

https://www.portshift.io/blog/kubernetes-multi-cluster-service-mesh/

https://www.portshift.io/blog/psp-kubernetes-security/

Q&A

Thank You

portshift