Kubernetes 1.31 - Security Highlights

What you need to know

Maxime Coquerel

Speaker

Maxime Coquerel

Principal Kubernetes Cloud Security - RBC

CISSP, CCSP, CSSK, Azure Security Engineer Associate

Email : max.coquerel@gmail.com

Blog: zigmax.net (since 2012)

Github: https://github.com/zigmax

X: @zig_max

MVP Microsoft® Most Valuable Professional

Disclaimer

“*Any views or opinions expressed in this presentation are those of the presenter and not necessarily represent the view and opinions of my employer, its ownership, management or its employees .*“

Kubernetes 1.31 - Release Note

- **Code Name: Elli**

Kubernetes v1.31's Elli is a cute and joyful dog, with a heart of gold and a nice sailor's cap, as a playful wink to the huge and diverse family of Kubernetes contributors.

Kubernetes v1.31 marks the first release after the project has successfully celebrated its first 10 years.

Reference: https://kubernetes.io/blog/2024/08/13/kubernetes-v1-31-release/

AppArmor Support (GA)

- **General Availability**: AppArmor support is now fully integrated.
- **Key Change**: Moved from **annotations** to the **securityContext** field.
- **Benefit**: Enhances workload security by allowing users to specify AppArmor profiles directly in pod manifests.
- **Impact**: Enables limiting application capabilities with predefined security policies for improved protection.

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: apparmor-pod
spec:
  containers:
  - name: my-app
    image: nginx:latest
    securityContext:
      apparmorProfile: runtime/default
```

Private Registry Image Pulling with Secrets

Use Case: **Securely pull container images from a private registry.**

Step 1: Create a secret for registry credentials.

- Command: kubectl create secret docker-registry

Step 2: Configure your pod to use the secret.

- Field: imagePullSecrets in the pod specification

Result: Kubernetes securely pulls images from the private registry using the stored credentials.

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: private-reg-pod
spec:
  containers:
  - name: my-app
    image: <your-registry-server>/my-app:latest
  imagePullSecrets:
  - name: my-registry-secret
```

Pod Level Resource Limits

Overview:

- Provides precise control over resource distribution.
- Defines maximum resource use for both containers and pods.

Key Features:

- **Container-Level Limits**: Sets specific resource ceilings for each container.
- **Pod-Level Limits**: Restricts total resource usage for all containers within a pod.

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: my-pod
spec:
  containers:
  - name: my-container
    image: my-image
    resources:
      limits:
        cpu: "2"
        memory: "2Gi"
  limits:
    cpu: "4"
    memory: "4Gi"
```

Restrictions on Anonymous API Access (Alpha)

Overview:

- Introduces restrictions on anonymous access to the Kubernetes API server.

Features:

- **Default Deny Policy**: Anonymous users are denied access to most API resources.
- **Controlled Access**: Specific resources can be explicitly allowed through RBAC (Role-Based Access Control).

Key Changes:

- Enhances security by limiting actions that unauthenticated users can perform.
- Reduces the risk of unauthorized access and potential exploitation.

Benefits:

- Improves overall cluster security by ensuring that only authenticated users can perform sensitive actions.
- Encourages best practices for managing API access.

Finer Grained Authorization Based on Selectors (Alpha)

Overview: Introduces finer-grained authorization using selectors for improved access control.

Key Features:

- Allows webhook authorizers and future in-tree authorizers to permit list and watch requests based on label and/or field selectors.

Benefit: Enhances security by enabling more precise control over access to resources.

```yaml
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: my-cluster-role
rules:
- apiGroups: ["*"]
  resources: ["pods"]
  verbs: ["list", "watch"]
  resourceAttributes:
    matchingLabels:
      app: my-app
```

Bound Service Account Token Improvements (Beta)

Key Features:

- **Bound Tokens**: Tokens are bound to a specific pod, reducing the risk of token theft.
- **Expiration**: Tokens have a limited lifetime, automatically expiring after a set duration.
- **Dynamic Token Generation**: Allows for on-demand generation of tokens for improved flexibility.

Benefits:

- **Increased Security**: Limits the impact of token exposure by restricting token usage to a specific pod and timeframe.
- **Simplified Management**: Reduces the need for manual token management, improving operational efficiency.
- **Enhanced Compliance**: Aligns with security best practices by enforcing short-lived credentials.

```yaml
apiVersion: v1
kind: ServiceAccount
metadata:
  name: my-service-account
annotations:
  serviceaccount.k8s.io/token-node-binding: "true"
```

Randomized Pod Selection Algorithm

Overview:

- Introduces a new algorithm for selecting pods in a more randomized manner when downscaling ReplicaSets.

How It Works:

- The algorithm randomly selects a pod from a set of eligible pods, rather than using a deterministic approach.
- This helps prevent bottlenecks by avoiding predictable patterns in pod selection.

Key Features:

- Enhances load balancing by distributing requests more evenly across pods.
- Reduces the likelihood of pod starvation, improving overall resource utilization.

Benefits:

- Improves application performance through better resource allocation.
- Increases resilience by ensuring that no single pod becomes a bottleneck.

Resources

Kubernetes Release Notes:

- Official release notes provide detailed information about all changes, enhancements, and bug fixes, including security features.
- Kubernetes 1.31 Release Notes

Security Updates and Advisories:

- Check for any security advisories issued for 1.31 to stay updated on vulnerabilities and fixes.
- Kubernetes Security Advisories

Kubernetes Security Documentation:

- This includes best practices for securing Kubernetes clusters, threat modeling, and compliance information.
- Kubernetes Security Overview

Kubernetes Enhancement Proposals (KEPs):

- KEPs that are relevant to security in 1.31 can provide insights into the rationale and implementation details of security features.
- Kubernetes KEPs

Questions?