KUBERNETES v1.25 WHAT YOU NEED TO KNOW

SENTHIL RAJA CHERMAPANDIAN

SPEAKER INTRO

Senthil Raja Chermapandian

Principal Software Engineer, Ericsson

Architecting Cloud-native AI/ML Platforms

Organizer: KCD Chennai 2022 (kcdchennai.in)

Maintainer: github.com/senthilrch/kube-fledged

Tech blogger: medium.com/@senthilrch

Twitter: @senthilrch

LinkedIn: linkedin.com/in/senthilrch

HIGHLIGHTS

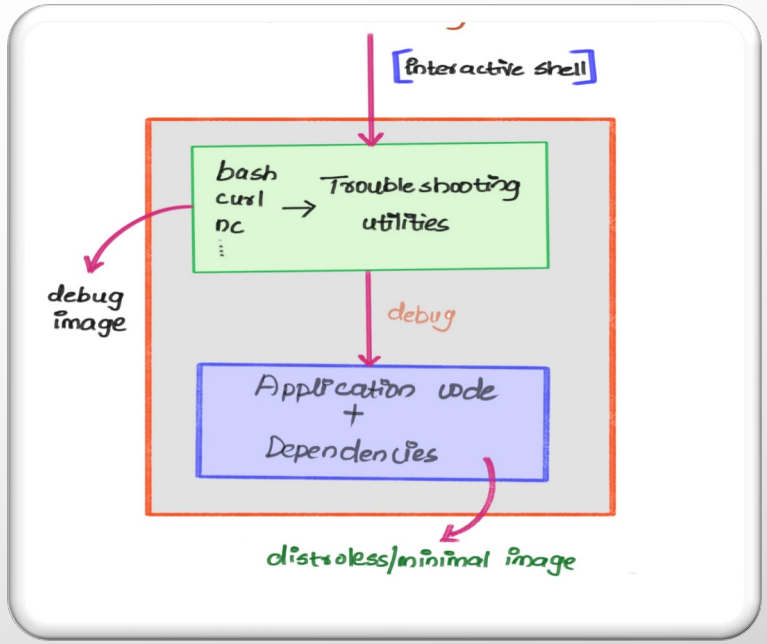
- This release includes a total of **40 Enhancements**
- PodSecurityPolicy is removed; Pod Security Admission graduates to Stable
- Ephemeral Containers Graduate to Stable
- Support for cgroups v2 Graduates to Stable
- Promoted endPort in Network Policy to Stable
- Promoted Local Ephemeral Storage Capacity Isolation to Stable
- Promoted CSI Ephemeral Volume to Stable
- Promoted CRD Validation Expression Language to Beta
- Introduced KMS v2 API

PodSecurityPolicy is removed; Pod Security Admission graduates to stable

- Podsecuritypolicy is a built-in admission controller that allows a cluster administrator to control security-sensitive aspects of the pod specification.
- Since kubernetes 1.3, PodSecurityPolicy has been the built-in way to control what sorts of settings are allowed in the resources defined in your cluster
- The way SSPs are applied to pods has proven confusing to nearly everyone that has attempted to use them. It is easy to **accidentally** grant broader permissions than intended, and **difficult** to inspect which PSP(s) apply in a given situation.
- Pod Security Admission
- Pod Security admission places requirements on a Pod's Security Context and other related fields according to the three levels defined by the Pod Security Standards: **privileged, baseline, and restricted**.
- Once the feature is enabled, you can configure namespaces to define the admission control mode you want to use for pod security in each namespace: **enforce, audit, warn**
- The Admission controller is configured using the **AdmissionConfiguration** API resource to set cluster-wide defaults and exemptions

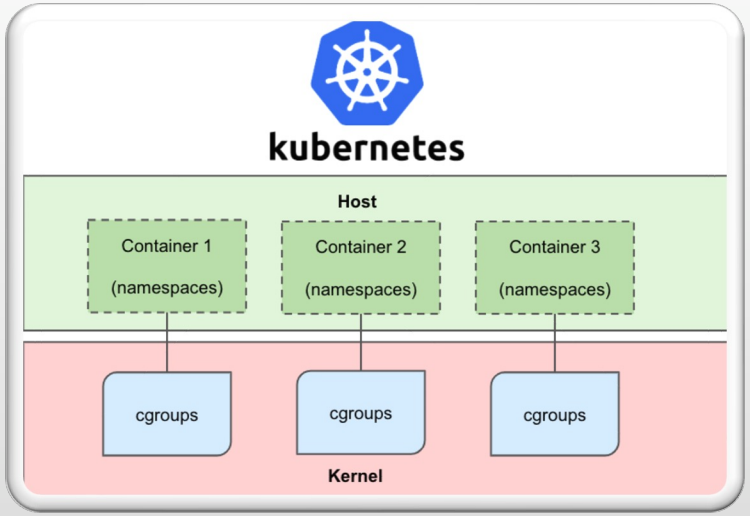
Ephemeral Containers Graduate to Stable

- Ephemeral containers are containers that exist for only a limited time within an existing pod.
- This is particularly useful for troubleshooting when you need to examine another container but cannot use kubectl exec because that container has crashed or its image lacks debugging utilities.
- Ephemeral containers graduated to beta in kubernetes v1.23, and with this release, the feature graduates to stable.
- Ephemeral containers are described using the same `containerspec` as regular containers, but many fields are incompatible and disallowed for ephemeral containers.
- Ephemeral containers are created using a special `ephemeralcontainers` handler in the api rather than by adding them directly to `pod.Spec`, so it's not possible to add an ephemeral container using `kubectl edit`.
- When using ephemeral containers, it's helpful to enable process namespace sharing so you can view processes in other containers.



Support for cgroups v2 Graduates to Stable

- It has been more than two years since the Linux kernel cgroups v2 API was declared stable.
- With some distributions now defaulting to this API, Kubernetes must support it to continue operating on those distributions.
- cgroups v2 offers several improvements over cgroups v1. cgroups v1 will continue to be supported
- On Linux, control groups constrain resources that are allocated to processes.
- The kubelet and the underlying container runtime need to interface with cgroups to enforce resource management for pods and containers which includes cpu/memory requests and limits for containerized workloads.
- cgroup v2 has the following requirements: OS distribution enables cgroup v2, Linux Kernel version is 5.8 or later; Container runtime supports cgroup v2. The kubelet and the container runtime are configured to use the systemd cgroup driver
- The kubelet automatically detects that the OS is running on cgroup v2 and performs accordingly with no additional configuration required.



Promoted endPort in Network Policy to Stable

- Promoted endPort in Network Policy to GA.
- Network Policy providers that support endPort field now can use it to specify a range of ports to apply a Network Policy. Previously, each Network Policy could only target a single port.
- Please be aware that endPort field must be supported by the Network Policy provider.
- If your provider does not support endPort, and this field is specified in a Network Policy, the Network Policy will be created covering only the port field (single port).

```
policyTypes:
  - Egress
egress:
  - to:
      - ipBlock:
          cidr: 10.0.0.0/24
    ports:
      - protocol: TCP
        port: 32000
        endPort: 32768
```

Promoted Local Ephemeral Storage Capacity Isolation to Stable

- The Local Ephemeral Storage Capacity Isolation feature moved to GA.
- This was introduced as alpha in 1.8, moved to beta in 1.10, and it is now a stable feature.
- It provides support for capacity isolation of local ephemeral storage between pods, such as EmptyDir, so that a pod can be hard limited in its consumption of shared resources by evicting Pods if its consumption of local ephemeral storage exceeds that limit.

```yaml
spec:
  containers:
  - name: app
    image: images.my-company.example/app:v4
    resources:
      requests:
        ephemeral-storage: "2Gi"
      limits:
        ephemeral-storage: "4Gi"
    volumeMounts:
    - name: ephemeral
      mountPath: "/tmp"
```

```yaml
volumes:
  - name: ephemeral
    emptyDir:
      sizeLimit: 500Mi
```

Promoted CSI Ephemeral Volume to Stable

- The CSI Ephemeral Volume feature allows CSI volumes to be specified directly in the pod specification for ephemeral use cases.
- They can be used to inject arbitrary states, such as configuration, secrets, identity, variables or similar information, directly inside pods using a mounted volume.
- This was initially introduced in 1.15 as an alpha feature, and it moved to GA.

```
volumes:
  - name: my-csi-inline-vol
    csi:
      driver: inline.storage.kubernetes.io
      volumeAttributes:
        foo: bar
```

Promoted CRD Validation Expression Language to Beta

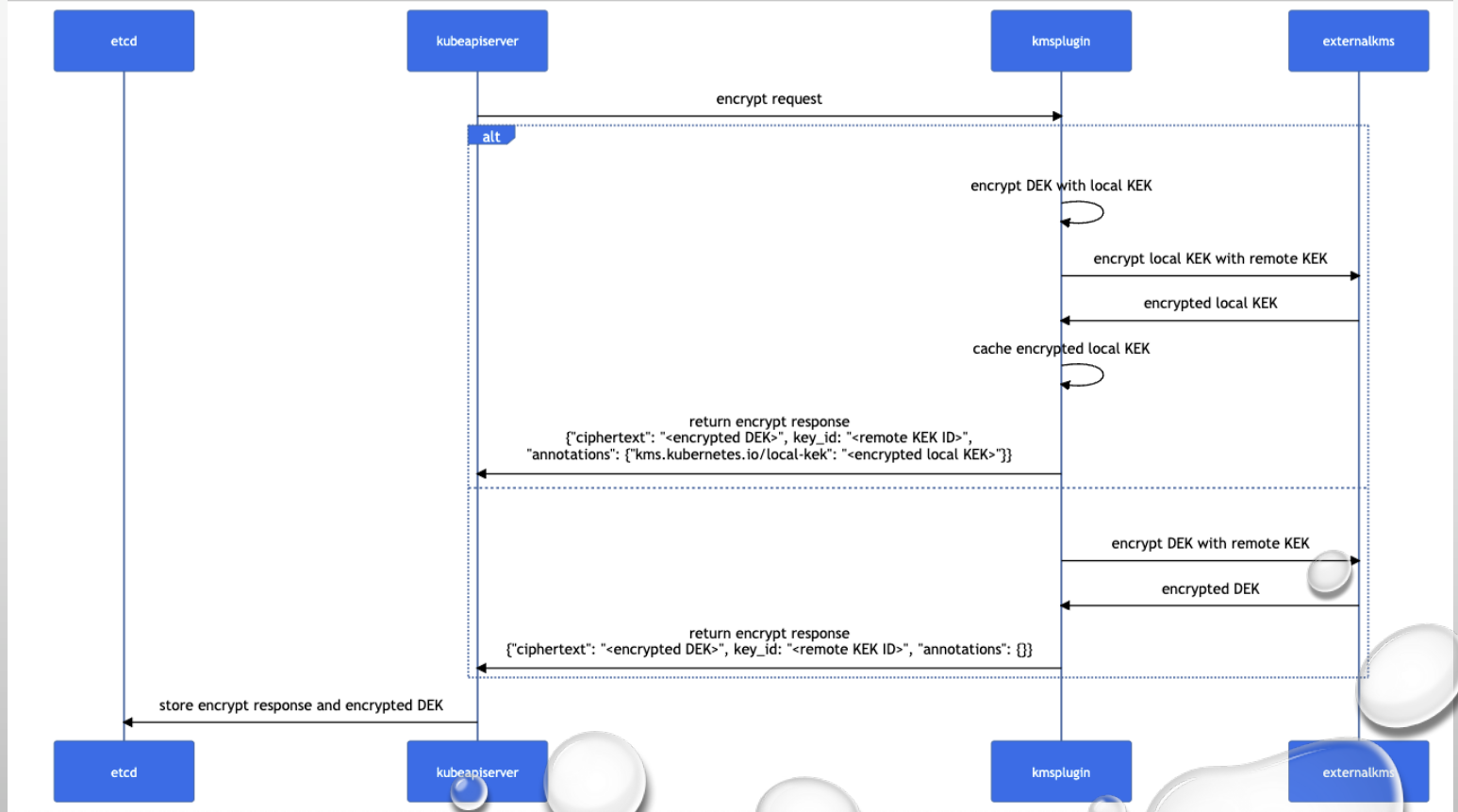
- CRD Validation Expression Language is promoted to beta, which makes it possible to declare how custom resources are validated using the Common Expression Language (CEL).

```yaml
apiVersion: apiextensions.k8s.io/v1
kind: CustomResourceDefinition
...
  schema:
    openAPIV3Schema:
      type: object
      properties:
        spec:
          x-kubernetes-validations:
            - rule: "self.minReplicas <= self.maxReplicas"
              message: "minReplicas cannot be larger than maxReplicas"
          type: object
          properties:
            minReplicas:
              type: integer
            maxReplicas:
              type: integer
```

Introduced KMS v2 API

- Introduce KMS v2alpha1 API to add performance, rotation, and observability improvements.

```yaml
apiVersion: apiserver.config.k8s.io/v1
kind: EncryptionConfiguration
resources:
  - resources:
      - secrets
      - configmaps
      - pandas.awesome.bears.example
    providers:
      - kms:
          apiVersion: v2
          name: myKmsPluginFoo
          endpoint: unix:///tmp/socketfile.sock
          cachesize: 100
          timeout: 3s
      - kms:
          name: myKmsPluginBar
          endpoint: unix:///tmp/socketfile.sock
          cachesize: 100
          timeout: 3s
```



RESOURCES

- POD SECURITY ADMISSION | KUBERNETES
- DEBUG RUNNING PODS | KUBERNETES
- ABOUT CGROUP V2 | KUBERNETES
- NETWORK POLICIES | KUBERNETES
- RESOURCE MANAGEMENT FOR PODS AND CONTAINERS | KUBERNETES
- EPHEMERAL VOLUMES | KUBERNETES
- https://github.com/kubernetes/enhancements/blob/master/keps/sig-api-machinery/2876-crd-validation-expression-language/README.md
- KUBERNETES 1.25: KMS V2 IMPROVEMENTS | KUBERNETES
- USING A KMS PROVIDER FOR DATA ENCRYPTION | KUBERNETES

THANK YOU

SENTHIL RAJA CHERMAPANDIAN