Deep Dive into Kubernetes Part 2

Imesh Gunaratne, WSO2

Agenda

- Docker Image Size Optimization
- Docker Image Size Optimization Demo
- Persistent Volumes
  - Static Volumes
  - Dynamic Volumes
- Container Security
  - File System Permissions
  - Pod Security Context
  - Pod Security Policies

Agenda Cont.

- Kubernetes on Google Cloud Platform
- GKE Demo
- Kubernetes on AWS
- Kubernetes on Azure

Docker Image Size Optimization

A Sample Dockerfile

```
FROM anapsix/alpine-java:8

ARG USER_GROUP_ID=1000
ARG USER_GROUP=bar
ARG USER_ID=2000
ARG USER=foo

RUN addgroup -S -g ${USER_GROUP_ID} ${USER_GROUP} && \
  adduser -S -g ${USER_GROUP_ID} -u ${USER_ID} ${USER}

COPY --chown=foo:bar helloworld-* /tmp/

USER ${USER_ID}
ENTRYPOINT [ "java", "-jar", "/tmp/helloworld-2.5.3-SNAPSHOT.jar" ]
```

Docker Image Size Optimization

1. Use a smaller base image

2. Install only application dependent software

3. Minimize layers and combine RUN commands

```
RUN groupadd --system -g ${USER_GROUP_ID} ${USER_GROUP} && \
    useradd --system --create-home --home-dir ${USER_HOME} --no-log-init
-g ${USER_GROUP_ID} -u ${USER_ID} ${USER}
```

4. List layers according to change frequency

Docker Image Size Optimization Cont.

5. Use --no-install-recommends on apt-get install:

```
RUN apt-get update && \
    apt-get install -y --no-install-recommends \
    <package-name>
```

6. Add rm -rf /var/lib/apt/lists/* to same layer as apt-get installs:

```
RUN apt-get update && \
    apt-get install -y --no-install-recommends \
    <package-name> \
    rm -rf /var/lib/apt/lists/*
```

Docker Image Size Optimization Cont.

7. Copy archive files after extracting them locally:

```
COPY ${FILES}/${JDK} ${USER_HOME}/java/
```

8. Use --chown key instead of chown command when copying files:

```
COPY --chown=wso2carbon:wso2 ${FILES}/${JDK} ${USER_HOME}/java/
```

Docker Image Size Optimization Cont.

Finally use docker history command to verify the image structure:

```
docker history gcr.io/google-samples/node-hello:1.0

IMAGE               CREATED             CREATED BY                                      SIZE
4c7ea8709739        2 years ago         /bin/sh -c #(nop) CMD ["/bin/sh" "-c" "node …   0B
<missing>           2 years ago         /bin/sh -c #(nop) COPY file:0ee96426ce9ede6e…   861B
<missing>           2 years ago         /bin/sh -c #(nop) EXPOSE 8080/tcp               0B
<missing>           2 years ago         /bin/sh -c #(nop) CMD ["node"]                  0B
<missing>           2 years ago         /bin/sh -c curl -SLO "https://nodejs.org/dis…   37MB
<missing>           2 years ago         /bin/sh -c #(nop) ENV NODE_VERSION=4.4.2        0B
<missing>           2 years ago         /bin/sh -c #(nop) ENV NPM_CONFIG_LOGLEVEL=in…   0B
<missing>           2 years ago         /bin/sh -c set -ex   && for key in     9554F…   60.4kB
<missing>           2 years ago         /bin/sh -c apt-get update && apt-get install…   315MB
<missing>           2 years ago         /bin/sh -c apt-get update && apt-get install…   123MB
<missing>           2 years ago         /bin/sh -c apt-get update && apt-get install…   44.3MB
<missing>           2 years ago         /bin/sh -c #(nop) CMD ["/bin/bash"]             0B
<missing>           2 years ago         /bin/sh -c #(nop) ADD file:b5391cb13172fb513…   125MB
```

Docker Image
Size Optimization Demo

Persistent Volumes

Persistent Volume Usage Design

Persistent Volume Provisioning Types

- **Static**
  - Manually map physical storage to PVs
- **Dynamic**
  - When static PVs are unavailable, K8S will check the availability of dynamic PVs
  - It dynamically creates volumes using storage provisioners
  - The storage provisioner may use a single physical storage for providing multiple dynamic PVs

https://kubernetes.io/docs/concepts/storage/persistent-volumes/
https://github.com/kubernetes-incubator/external-storage/

Persistent Volume Claims

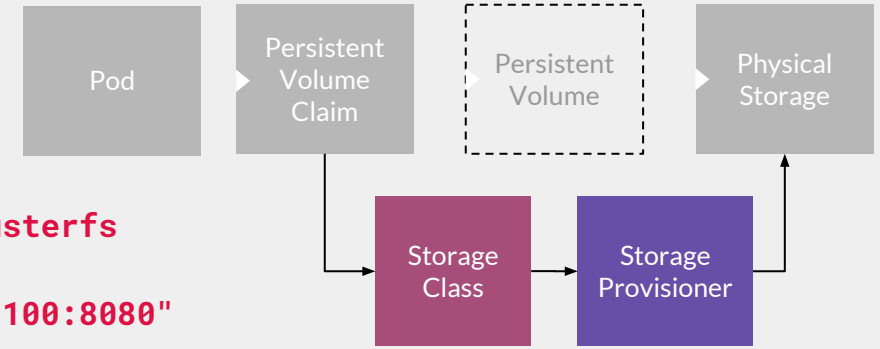
```yaml
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: myclaim
spec:
  accessModes:
    - ReadWriteOnce
  volumeMode: Filesystem
  resources:
    requests:
      storage: 8Gi
  storageClassName: slow
  selector:
    matchLabels:
      release: "stable"
    matchExpressions:
      - {key: environment, operator: In, values: [dev]}
```

Static Persistent Volumes

```yaml
apiVersion: v1
kind: PersistentVolume
metadata:
  name: pv0003
spec:
  capacity:
    storage: 5Gi
  volumeMode: Filesystem
  accessModes:
    - ReadWriteOnce
  persistentVolumeReclaimPolicy: Recycle
  storageClassName: slow
  mountOptions:
    - hard
    - nfsvers=4.1
  nfs:
    path: /tmp
    server: 172.17.0.2
```

Dynamic Persistent Volumes

```yaml
kind: StorageClass
apiVersion: storage.k8s.io/v1
metadata:
  name: gluster-vol-default
provisioner: kubernetes.io/glusterfs
parameters:
  resturl: "http://192.168.10.100:8080"
  restuser: ""
  secretNamespace: ""
  secretName: ""
allowVolumeExpansion: true
```



Persistent Volume Volume Modes

- Filesystem
- Block

```yaml
apiVersion: v1
kind: PersistentVolume
metadata:
  name: block-pv
spec:
  capacity:
    storage: 10Gi
  accessModes:
    - ReadWriteOnce
  volumeMode: Block
  persistentVolumeReclaimPolicy: Retain
  fc:
    targetWWNs: ["50060e801049cfd1"]
    lun: 0
    readOnly: false
```

https://kubernetes.io/docs/concepts/storage/persistent-volumes/#volume-modes

Persistent Volume Access Modes

- **ReadWriteOnce**
  - read-write by a single node
- **ReadOnlyMany**
  - read-only by many nodes
- **ReadWriteMany**
  - read-write by many nodes

https://kubernetes.io/docs/concepts/storage/persistent-volumes/#access-modes

Persistent Volume Binding

- Binding a PVC to PV will be done based on requested:
  - Amount of storage
  - Access mode
  - Storage Class (Optional)

https://kubernetes.io/docs/concepts/storage/persistent-volumes/

Persistent Volume Re-claim Policies

- **Retain**
  - Manual reclamation
- **Recycle**
  - Basic scrub (rm -rf /thevolume/*) on the volume and makes it available again for a new claim
- **Delete**
  - Deletion removes both the PV object from Kubernetes, as well as the associated storage asset in the external infrastructure

https://kubernetes.io/docs/concepts/storage/persistent-volumes/

Container Security

Container Security

- Container host operating system security
- Container operating system security
- Application security
- Container registry permissions
- Network isolation
- Container filesystem permissions
- Persistent volume permissions
- Container/container cluster manager API endpoint permissions

File System Permission Management

File System Permissions

1. Create an OS user and a group in the container image:

```
RUN groupadd --system -g ${USER_GROUP_ID} ${USER_GROUP} && \
   useradd --system --create-home --home-dir ${USER_HOME} --no-log-init -g
${USER_GROUP_ID} -u ${USER_ID} ${USER}
```

2. Specify user and the group for granting permissions to the filesystem:

```
COPY --chown=user:group ${FILES}/${JDK} ${USER_HOME}/java/
```

File System Permissions Cont.

3. Define the user id in the container image before the entrypoint:

```
USER ${USER_ID}
...
ENTRYPOINT ${SERVER_HOME}/bin/server.sh
```

Pod Security Context

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: security-context-demo
spec:
  securityContext:
    runAsUser: 1000
    fsGroup: 2000
  volumes:
  - name: sec-ctx-vol
    emptyDir: {}
  containers:
  - name: sec-ctx-demo
    image: gcr.io/google-samples/node-hello:1.0
    volumeMounts:
    - name: sec-ctx-vol
      mountPath: /data/demo
    securityContext:
      allowPrivilegeEscalation: false
```

Define the user id and the group id in the pod under security context:

https://kubernetes.io/docs/tasks/configure-pod-container/security-context/

Persistent Volume Permissions

- By default PVs would get root permissions (root:root).
- If the container is designed to run using a non-root user, that user may not have permissions for accessing the PVs.
- Therefore, the user group (gid) used in the container may need to be used for granting access to PVs.
- This can be done by:
  - Defining the gid in the PV.
  - By using a Pod Security Policy and defining the gid using the fsGroup property.

Define Group ID in Persistent Volume

Define the group id specified in the container image in the PV:

```
kind: PersistentVolume
apiVersion: v1
metadata:
  name: pv1
  annotations:
    pv.beta.kubernetes.io/gid: "1234"
```

https://kubernetes.io/docs/tasks/configure-pod-container/configure-persistent-volume-storage/

Pod Security Policies

Control Aspect	Field Names
Running of privileged containers	`privileged`
Usage of the root namespaces	`hostPID`, `hostIPC`
Usage of host networking and ports	`hostNetwork`, `hostPorts`
Usage of volume types	`volumes`
Usage of the host filesystem	`allowedHostPaths`
White list of FlexVolume drivers	`allowedFlexVolumes`
Allocating an FSGroup that owns the pod's volumes	`fsGroup`

https://kubernetes.io/docs/concepts/policy/pod-security-policy/

Pod Security Policies Cont.

Control Aspect	Field Names
Requiring the use of a read only root file system	`readOnlyRootFilesystem`
The user and group IDs of the container	`runAsUser`, `supplementalGroups`
Restricting escalation to root privileges	`allowPrivilegeEscalation`, `defaultAllowPrivilegeEscalation`
Linux capabilities	`defaultAddCapabilities`, `requiredDropCapabilities`, `allowedCapabilities`

https://kubernetes.io/docs/concepts/policy/pod-security-policy/

Pod Security Policies Cont.

Control Aspect	Field Names
The SELinux context of the container	`seLinux`
The AppArmor profile used by containers	`annotations`
The seccomp profile used by containers	`annotations`
The sysctl profile used by containers	`annotations`

https://kubernetes.io/docs/concepts/policy/pod-security-policy/

Defining File System Group (fsGroup)

```yaml
apiVersion: policy/v1beta1
kind: PodSecurityPolicy
metadata:
  name: restricted
  annotations:
    ...
spec:
  ...
  fsGroup:
    rule: 'MustRunAs'
    ranges:
      # Allow following gid range for accessing PVs
      - min: 1000
        max: 1000
```

Defining Run As User, Supplemental Groups

```
apiVersion: policy/v1beta1
kind: PodSecurityPolicy
metadata:
  name: example
spec:
  ...
  # start container using
following uid and gid range:
  runAsUser:
    rule: MustRunAs
    ranges:
  supplementalGroups:
    rule: RunAsAny
  ...
```

- **runAsUser**
  - MustRunAs (uid range)
  - MustRunAsNonRoot
  - RunAsAny
- **supplementalGroups:**
  - MustRunAs (gid range)
  - RunAsAny

Kubernetes on Google Cloud Platform

Kubernetes on Google Cloud

- It’s called Google Kubernetes Engine (GKE)
- Kubernetes is provided as a service
  - Masters are provided for free
- IAM: Role based access with Google Accounts
- Private container registries
- Autoscaling
- Auto upgrade Kubernetes clusters
- Logging and Monitoring with Stackdriver

https://cloud.google.com/kubernetes-engine/

Kubernetes on Google Cloud Cont.

- **Databases:**
  - Cloud SQL: MySQL, PostgreSQL
- **Persistent Volumes:**
  - **GCEPersistentDisk**: ReadWriteOnce, ReadOnlyMany
  - **NFS, Glusterfs**: ReadWriteMany
- **Load Balancers:**
  - Ingresses/Load Balancer Type Services creates Google Cloud Load Balancers

https://cloud.google.com/kubernetes-engine/

GKE Demo

Kubernetes on AWS

Kubernetes on AWS

- Recently announced: Amazon Elastic Container Service (EKS)
- Kubernetes will be provided as a service with EKS
  - Masters will be provided for free
- AWS Auth for role based access
- Currently, KOPs can be used for creating a self managed Kubernetes clusters on AWS

https://aws.amazon.com/eks/

Kubernetes on AWS

- **Databases:**
  - RDS: Amazon Aurora, MySQL, PostgreSQL, MariaDB, Oracle, MS SQL
- **Persistent Volumes:**
  - **EBS**: ReadWriteOnce
  - **EFS, NFS, Glusterfs** -> ReadWriteMany
- **Load Balancers:**
  - Ingresses/Load Balancer Type Services creates AWS Load Balancers

https://aws.amazon.com/eks/

Kubernetes on Azure

Kubernetes on Azure

- Azure Container Service (AKS)
- Currently in preview mode
- A managed Kubernetes service:
  - Masters will be provided for free
- Private container registries
- Autoscaling
- Automated Kubernetes cluster upgrades

https://azure.microsoft.com/en-us/services/container-service/

Kubernetes on Azure

- **Databases:**
  - Azure SQL: MySQL, PostgreSQL, MS SQL
- **Persistent Volumes:**
  - **AzureDisk**-> ReadWriteOnce
  - **AzureFile, NFS, Glusterfs**: ReadWriteOnce, ReadOnlyMany, ReadWriteMany
- **Load Balancers:**
  - Ingresses/Load Balancer Type Services creates Azure Load Balancers

https://azure.microsoft.com/en-us/services/container-service/

Questions & Feedback

References

References

- Tips to Reduce Docker Image Sizes:
  - https://hackernoon.com/tips-to-reduce-docker-image-sizes-876095da3b34
- Introduction to Container Security:
  - https://www.docker.com/sites/default/files/WP_IntrotoContainerSecurity_08.19.2016.pdf
- Ten Layers of Container Security:
  - https://www.redhat.com/cms/managed-files/cl-container-security-openshift-cloud-devops-tech-detail-f7530kc-201705-en.pdf

References Cont.

- Kubernetes Persistent Volumes:
  - https://kubernetes.io/docs/concepts/storage/persistent-volumes/
- Kubernetes External Storage Plugins, Provisioners, and Helper Libraries:
  - https://github.com/kubernetes-incubator/external-storage/
- Pod Security Policies:
  - https://kubernetes.io/docs/concepts/policy/pod-security-policy/

Thank You!