Registry Inside a Cluster

Create Registry in Cluster

Define Local Storage

> vi localStorage.yml

localStorage.yml

```
apiVersion: v1
kind: PersistentVolume
metadata:
  name: local-storage
spec:
  capacity:
    storage: 10Gi
  # volumeMode field requires BlockVolume Alpha feature gate to be enabled.
  volumeMode: Filesystem
  accessModes:
  - ReadWriteOnce
  persistentVolumeReclaimPolicy: Delete
  storageClassName: local-storage
  local:
    path: /var/k8s/LOCAL_STORAGE
  nodeAffinity:
    required:
      nodeSelectorTerms:
      - matchExpressions:
        - key: kubernetes.io/hostname
          operator: In
          values:
          - k8sworker1
          - k8sworker2
          - k8sworker3
          - docker-for-desktop

---
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: local-storage-claim
spec:
  storageClassName: local-storage
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 3Gi
```

For running on a docker for desktop cluster, you will probably need to update the path to a folder under the users home directory.

ie.
path: /Users/john.mehan/k8s/LOCAL_STORAGE

Apply the yml file

> *kubectl apply -f localStorage.yml*

On each of the worker nodes, create the folder specified in 'path'.

> *ssh k8sworker1*
>
> *sudo mkdir -p /var/k8s/LOCAL_STORAGE*

Repeat for all worker nodes.

Create Password File

We will generate a password file to use with our registry. The default username password will be test/testpw.

> *vi htpasswdGenerator.yml*

htpasswdGenerator.yml

```
apiVersion: v1
kind: Pod
metadata:
  name: htpasswd-generator
spec:
  containers:
  - name: htpasswd-generator
    image: registry:2
    command: ["/usr/bin/htpasswd"]
    args: ["-Bcb", "/auth/htpasswd", "test", "testpw"]
    volumeMounts:
        - mountPath: /auth
          name: local-vol
          subPath: registry/auth
  affinity:
    nodeAffinity:
      requiredDuringSchedulingIgnoredDuringExecution:
        nodeSelectorTerms:
        - matchExpressions:
          - key: kubernetes.io/hostname
            operator: In
            values:
            - k8sworker2
            - docker-for-desktop
  volumes:
  - name: local-vol
    persistentVolumeClaim:
      claimName: local-storage-claim
  restartPolicy: OnFailure
```

> *kubectl apply -f htpasswdGenerator.yml*

Create SSL Certificate

Create CSR

We are using CFSSL to generate the Certificate Signing Request.

Install CFSSL on your mac using the following command:

> *brew install cfssl*

Create script to create CSR:

> *vi createCSR.sh*

createCSR.sh

```
cat <<EOF | cfssl genkey - | cfssljson -bare server
{
  "hosts": [
    "registry-ext.default.svc.cluster.local",
    "registry.default.svc.cluster.local",
    "registry.default.pod.cluster.local"
  ],
  "CN": "registry.default.pod.cluster.local",
  "key": {
    "algo": "ecdsa",
    "size": 256
  }
}
EOF
```

Run script:

> *./createCSR*

This will generate two files:

- server-key.pem
- server.csr

Submit CSR to Kubernetes Cluster for Approval

Login to the master node

Create script to submit CSR

> *vi submitCSR.sh*

submitCSR.sh

```
cat <<EOF | kubectl apply -f -
apiVersion: certificates.k8s.io/v1beta1
kind: CertificateSigningRequest
metadata:
  name: registry.default
spec:
  groups:
  - system:authenticated
  request: $(cat server.csr | base64 | tr -d '\n')
  usages:
  - digital signature
  - key encipherment
  - server auth
EOF

kubectl get csr
```

Run the script:

> *./submitCSR.sh*

```
certificatesigningrequest.certificates.k8s.io/registry.default created
```

Approve CSR

> *kubectl certificate approve registry.default*
>
> *kubectl get csr*

```
NAME               AGE   REQUESTOR          CONDITION
registry.default   30s   kubernetes-admin   Approved,Issued
```

Get Signed Certificate

> *kubectl get csr registry.default -o jsonpath='{.status.certificate}' | base64 --decode > server.crt*

Create Secret for SSL Certificate

> *kubectl create secret generic registry-ssl --from-file=./server.crt --from-file=./server-key.pem*
>
> *kubectl get secrets*

```
NAME                  TYPE                                  DATA   AGE
registry-ssl          Opaque                                2      32s
```

Create Registry Pod and Service

We will create our registry and tie it to node k8sworker2. See nodeSelector in following registry.yml

> *vi registry.yml*

registry.yml

```
apiVersion: v1
kind: Pod
metadata:
  name: registry
  labels:
    app: registry
spec:
  containers:
  - name: registry
    image: registry:2
    env:
    - name: REGISTRY_AUTH
      value: htpasswd
    - name: REGISTRY_AUTH_HTPASSWD_REALM
      value: "Registry Realm"
    - name: REGISTRY_AUTH_HTPASSWD_PATH
      value: /auth/htpasswd
    - name: REGISTRY_HTTP_ADDR
      value: 0.0.0.0:5000
    - name: REGISTRY_HTTP_TLS_CERTIFICATE
      value: /ssl/server.crt
    - name: REGISTRY_HTTP_TLS_KEY
      value: /ssl/server-key.pem
    ports:
    - containerPort: 5000
    volumeMounts:
    - mountPath: /auth
      name: local-vol
      subPath: registry/auth
    - mountPath: /var/lib/registry
      name: local-vol
      subPath: registry/data
    - mountPath: /ssl
      name: registry-ssl
      readOnly: true
  affinity:
    nodeAffinity:
      requiredDuringSchedulingIgnoredDuringExecution:
        nodeSelectorTerms:
        - matchExpressions:
          - key: kubernetes.io/hostname
            operator: In
            values:
            - k8sworker2
            - docker-for-desktop
  volumes:
  - name: local-vol
    persistentVolumeClaim:
      claimName: local-storage-claim
  - name: registry-ssl
    secret:
      secretName: registry-ssl
---
apiVersion: v1
kind: Service
metadata:
  name: registry-ext
spec:
  type: NodePort
  selector:
    app: registry
  ports:
    - port: 5000
      nodePort: 30500
      name: registry-ext
```

> *kubectl apply -f registry.yml*

Verify that the registry was successfully installed by issuing the following command:

> *curl -k --user test:testpw https://<HOST_OR_IP>:30500/v2/_catalog*

```
{"repositories":[]}
```

Create a secret in the Cluster for the Registry

Create a secret in the cluster that holds your authorization token

> *kubectl create secret docker-registry regcred --docker-server='https://127.0.0.1:30500/' --docker-username='test' --docker-password='testpw' --docker-email='test@irdeto.com'*

Feed the Registry

From user machine, revise the docker daemon file.

On Mac:

vi ~/.docker/daemon.json

```
{
  "debug" : true,
  "experimental" : false,
  "insecure-registries" : ["172.20.233.181:30500"]
}
```

Add an entry for insecure-registries for your cluster registry.

docker tag <image> <node_ip>:30500/<image>

docker push <node_ip>:30500/<image>

example:

> *docker tag nginx-jmehan:latest 172.20.233.181:30500/nginx-jmehan:latest*
>
> *docker push 172.20.233.181:30500/nginx-jmehan:latest*

Verify that the registry was successfully installed by issuing the following command:

> *curl -k --user test:testpw https://172.20.233.181:30500/v2/_catalog*

```
{"repositories":["nginx-jmehan"]}
```

Revise deployment.yaml files

Add imagePullSecrets section to your deployment yaml files:

imagePullSecrets:
- name: regcred

example:

Example: nginx pulled from registry

```
apiVersion: v1
kind: Pod
metadata:
  name: sample
  labels:
    app: sample
spec:
  containers:
  - name: sample
    image: 127.0.0.1:30500/nginx-jmehan:latest
    ports:
    - containerPort: 80
  imagePullSecrets:
      - name: regcred
```

Test that you can pull from the cluster registry

Create a custom docker image

> *mkdir html*
>
> *vi html/index.html*

```
<html>
<body>
<h1>Hello from John</h1>
</body>
</html>
```

> *vi Dockerfile*

```
FROM nginx
COPY html /usr/share/nginx/html
```

> *docker build -t nginx-jmehan .*

```
Sending build context to Docker daemon  5.632kB
Step 1/2 : FROM nginx
 ---> 7042885a156a
Step 2/2 : COPY html /usr/share/nginx/html
 ---> Using cache
 ---> b284da22e8cb
Successfully built b284da22e8cb
Successfully tagged nginx-jmehan:latest
```

Push the Custom Image to the new Registry

> *docker tag nginx-jmehan:latest 172.20.233.181:30500/nginx-jmehan:latest*
>
> *docker push 172.20.233.181:30500/nginx-jmehan:latest*

Verify that it is in the registry

> *curl -k --user test:testpw https://172.20.233.181:30500/v2/_catalog*

```
{"repositories":["nginx-jmehan"]}
```

Configure a node that uses the new image

> *vi sample.yml*

sample.yml

```
apiVersion: v1
kind: Pod
metadata:
  name: sample
  labels:
    app: sample
spec:
  containers:
  - name: sample
    image: 127.0.0.1:30500/nginx-jmehan:latest
    ports:
    - containerPort: 80
  imagePullSecrets:
      - name: regcred
---
apiVersion: v1
kind: Service
metadata:
  name: sample
spec:
  type: NodePort
  selector:
    app: sample
  ports:
    - port: 80
      nodePort: 30080
      name: sample
```

> *kubectl apply -f sample.yml*

Navigate to http://localhost:30080/ to see our new pod.

Troubleshooting

Start a ubuntu pod

> *kubectl run -it --image=ubuntu:latest bash*

From bash shell of new pod:

> *apt-get update*
>
> *apt-get install curl*
>
> *curl -k --user test:testpw https://registry:5000/v2/_catalog*

{"repositories":["nginx-jmehan"]}

References

Reference	URL
Deploy a registry server	https://docs.docker.com/registry/deploying/
Configuring a registry	https://docs.docker.com/registry/configuration/
Manage TLS Certificates in a Cluster	https://kubernetes.io/docs/tasks/tls/managing-tls-in-a-cluster/#requesting-a-certificate
Docker registry:2 Setup with TLS, Basic Auth, and persistent data	https://medium.com/@ManagedKube/docker-registry-2-setup-with-tls-basic-auth-and-persistent-data-8b98a2a73eec
Docker Login	https://docs.docker.com/engine/reference/commandline/login/
Using a Private Registry	https://kubernetes.io/docs/concepts/containers/images/#using-a-private-registry
Define a Command and Arguments for a Container	https://kubernetes.io/docs/tasks/inject-data-application/define-command-argument-container/
In Cluster Docker Regisity with TLS	https://medium.com/@jmarhee/in-cluster-docker-registry-with-tls-on-kubernetes-758eecfe8254