CKA Curriculum Part 4 - Cluster

Understand the Kubernetes cluster upgrade process

Kubeadm is likely going to be the tool of choice to upgrade a Kubernetes cluster. Firstly, define the version you wish to upgrade to, or find the latest stable release:

```
export VERSION=$(curl -sSL https://dl.k8s.io/release/stable.txthttps://dl.k8s.io/release/stable.txt)
export ARCH=amd64
```

Download the latest version of kubeadm

```
curl -sSL https://dl.k8s.io/release/${VERSION}/bin/linux/${ARCH}/kubeadmhttps://dl.k8s.io/release/${VERSION}/bin/linux/${ARCH}/kubeadm > kubeadm
```

Install Kubeadm

```
sudo install -o root -g root -m 0755 ./kubeadm /usr/bin/kubeadm
```

Alternatively, use apt

```
apt-mark unhold kubeadm && \
apt-get update && apt-get install -y kubeadm=1.14.x-00 && \
apt-mark hold kubeadm
```

Then, execute a upgrade plan

```
sudo kubeadm upgrade plan

Components that must be upgraded manually after you have upgraded the control plane with 'kubeadm upgrade apply':
COMPONENT   CURRENT       AVAILABLE
Kubelet     3 x v1.13.0   v1.14.1

Upgrade to the latest stable version:

COMPONENT            CURRENT   AVAILABLE
API Server           v1.13.5   v1.14.1
Controller Manager   v1.13.5   v1.14.1
Scheduler            v1.13.5   v1.14.1
Kube Proxy           v1.13.5   v1.14.1
CoreDNS              1.2.6     1.3.1
Etcd                 3.2.24    3.3.10

You can now apply the upgrade by executing the following command:

       kubeadm upgrade apply v1.14.1

As the output implies, execute the kubeadm upgrade command (under root/sudo) to apply the changes

linu
[preflight] Running pre-flight checks.
[upgrade] Making sure the cluster is healthy:
[upgrade/config] Making sure the configuration is correct:
[upgrade/config] Reading configuration from the cluster...
[upgrade/config] FYI: You can look at this config file with 'kubectl -n kube-system get cm kubeadm-config -oyaml'
[upgrade/version] You have chosen to change the cluster version to "v1.14.1"
[upgrade/versions] Cluster version: v1.13.5
[upgrade/versions] kubeadm version: v1.14.1
[upgrade/confirm] Are you sure you want to proceed with the upgrade? [y/N]: y
[upgrade/prepull] Will prepull images for components [kube-apiserver kube-controller-manager kube-scheduler etcd]
[upgrade/prepull] Prepulling image for component etcd.
```

After which, kubeadm will start pulling down updated container images and deploy those to your cluster.

If you run a kubectl get nodes on your cluster following a kubeadm upgrade apply command, you'll notice that the version listed hasn't changed:

```
NAME            STATUS   ROLES    AGE   VERSION
k8s-master-03   Ready    master   22h   v1.13.0
k8s-worker-03   Ready    <none>   22h   v1.13.0
k8s-worker-04   Ready    <none>   22h   v1.13.0
```

Recall from the kubeadm upgrade command:

```
Components that must be upgraded manually after you have upgraded the control plane with 'kubeadm upgrade apply':
COMPONENT   CURRENT       AVAILABLE
Kubelet     3 x v1.13.0   v1.14.1
```

The version number from kubectl get nodes refers to the kubelet version. As the tool implies, you need to upgrade this manually.

The following needs to be done on each node. Note the kubelet service needs restarting after the new binary has been implemented.

```
export VERSION=$(curl -sSL https://dl.k8s.io/release/stable.txthttps://dl.k8s.io/release/stable.txt)
export ARCH=amd64
curl -sSL https://dl.k8s.io/release/${VERSION}/bin/linux/${ARCH}/kubelethttps://dl.k8s.io/release/${VERSION}/bin/linux/${ARCH}/kubelet > kubelet
sudo install -o root -g root -m 0755 ./kubelet /usr/bin/kubelet
sudo systemctl restart kubelet.service
```

Alternatively, use apt:

```
apt-mark unhold kubelet && \
apt-get update && apt-get install -y kubelet=1.14.x-00 kubectl=1.14.x-00 && \
apt-mark hold kubelet
```

Facilitate Operating System Upgrades

Occasionally upgrades have to be made to the underlying host that's facilitating your pods. To do this in a graceful way, we can do the following.

Get a list of nodes running on which node

```
kubectl get pods -o wide

Drain the pods from the node:

kubectl drain k8s-worker04 --ignore-daemonsets

node/k8s-worker-04 cordoned
WARNING: Ignoring DaemonSet-managed pods: kube-flannel-ds-amd64-hq4wh, kube-proxy-cqpst
pod/nginx-7db9fccd9b-tdpx7 evicted
pod/coredns-fb8b8dccf-hj5km evicted
node/k8s-worker-04 evicted
```

As we can see, the non daemonset pods have been evicted and the host has been cordoned. This will apply the "SchedulingDisabled" flag on the node

```
kubectl get nodes
NAME          STATUS
k8s-worker-04   Ready,SchedulingDisabled
```

At this point, complete the required maintenance activities. Even if you reboot, the node will still be in a "SchedulingDisabled" state.

To release the node back to scheduling duties, uncordon the node:

```
kubectl uncordon k8s-worker-04
```

At which point it will be ready:

```
kubectl get nodes
NAME          STATUS
k8s-worker-04   Ready
```

Implement backup and restore methodologies

In a Kubernetes cluster there are two main pieces of data that need backing up:

- Certificate Files
- Etcd database

Backing up certificate files

Back up the /etc/kubernets/pki directory. This can also be done as a cronjob, and contains the following files:

```
/etc/kubernetes/pki$ ls
apiserver.crt
apiserver.key
ca.crt
front-proxy-ca.crt
front-proxy-client.key
apiserver-etcd-client.crt
apiserver-kubelet-client.crt
ca.key
front-proxy-ca.key
sa.key
apiserver-etcd-client.key
apiserver-kubelet-client.key
etcd front-proxy-client.crt
Sa.pub
```

Backing up etcd

Take a snapshot of the DB, then store it in a safe location

```
ETCDCTL_API=3 etcdctl snapshot save snapshot.db --cacert /etc/kubernetes/pki/etcd/server.crt --cert /etc/kubernetes/pki/etcd/ca.crt --key /etc/kubernetes/pki/etcd/ca.key
```

Verify the backup:

```
sudo ETCDCTL_API=3 etcdctl --write-out=table snapshot status snapshot.db
+----------+----------+------------+------------+
|   HASH   | REVISION | TOTAL KEYS | TOTAL SIZE |
+----------+----------+------------+------------+
| 2125d542 |   364069 |        770 |   3.8 MB |
+----------+----------+------------+------------+
```