Exercise 4.1: Basic Node Maintenance

In this section we will backup the **etcd** database then update the version of Kubernetes used on control plane nodes and worker nodes.

Backup The etcd Database

While the upgrade process has become stable, it remains a good idea to backup the cluster state prior to upgrading. There are many tools available in the market to backup and manage etcd, each with a distinct backup and restore process. We will use the included snapshot command, but be aware the exact steps to restore will depend on the tools used, the version of the cluster, and the nature of the disaster being recovered from.

1. Find the data directory of the **etcd** daemon. All of the settings for the pod can be found in the manifest.

```
student@cp:˜$ sudo grep data-dir /etc/kubernetes/manifests/etcd.yaml
```

```
    - --data-dir=/var/lib/etcd
```

2. Log into the **etcd** container and look at the options **etcdctl** provides. Use tab to complete the container name.

```
student@cp:˜$ kubectl -n kube-system exec -it etcd-<Tab> -- sh
```

On Container

(a) View the arguments and options to the **etcdctl** command. Take a moment to view the options and arguments available. As the Bourne shell does not have may features it may be easier to copy/paste the majority of the command and arguments after typing them out the first time.

```
# etcdctl -h
```

```
NAME:
        etcdctl - A simple command line client for etcd3.

USAGE:
        etcdctl [flags]
<output_omitted>
```

(b) In order to use TLS, find the three files that need to be passed with the **etcdctl** command. Change into the directory and view available files. Newer versions of **etcd** image have been minimized. As a result you may no longer have the **find** command, or really most commands. One must remember the URL /etc/kubernetes/pki/etcd. As the **ls** command is also missing we can view the files using **echo** instead.

```
# cd /etc/kubernetes/pki/etcd
```

```
# echo *
```

```
ca.crt ca.key healthcheck-client.crt healthcheck-client.key
peer.crt peer.key server.crt server.key
```

(c) Typing out each of these keys, especially in a locked-down shell can be avoided by using an environmental parameter. Log out of the shell and pass the various paths to the necessary files.

```
# exit
```

3. Check the health of the database using the loopback IP and port 2379. You will need to pass then peer cert and key as well as the Certificate Authority as environmental variables. The command is commented, you do not need to type out the comments or the backslashes.

```
student@cp:~$ kubectl -n kube-system exec -it etcd-k8scp -- sh \   #Same as before
 -c "ETCDCTL_API=3 \    #Version to use
 ETCDCTL_CACERT=/etc/kubernetes/pki/etcd/ca.crt \    #Pass the certificate authority
 ETCDCTL_CERT=/etc/kubernetes/pki/etcd/server.crt \    #Pass the peer cert and key
 ETCDCTL_KEY=/etc/kubernetes/pki/etcd/server.key \
 etcdctl endpoint health"     #The command to test the endpoint
```

```
https://127.0.0.1:2379 is healthy: successfully committed proposal: took = 11.942936ms
```

4. Determine how many databases are part of the cluster. Three and five are common in a production environment to provide 50%+1 for quorum. In our current exercise environment we will only see one database. Remember you can use `up-arrow` to return to the previous command and edit the command without having to type the whole command again. The command uses relative paths to the pki files for more clarity on the page.

```
student@cp:~$ kubectl -n kube-system exec -it etcd-k8scp -- sh -c \
 "ETCDCTL_API=3 --cert=./peer.crt --key=./peer.key --cacert=./ca.crt \
  etcdctl --endpoints=https://127.0.0.1:2379 member list"
```

```
fb50b7ddbf4930ba, started, k8scp, https://10.128.0.35:2380,
https://10.128.0.35:2379, false
```

5. You can also view the status of the cluster in a table format, among others passed with the **-w** option.

```
student@cp:~$ kubectl -n kube-system exec -it etcd-k8scp -- sh -c "ETCDCTL_API=3 \
 ETCDCTL_CACERT=/etc/kubernetes/pki/etcd/ca.crt ETCDCTL_CERT=/etc/kubernetes/pki/etcd/server.crt \
 ETCDCTL_KEY=/etc/kubernetes/pki/etcd/server.key  \
 etcdctl --endpoints=https://127.0.0.1:2379 member list -w table"
```

```
+------------------+---------+-------+--------------------------+--------------------------+------------+
|        ID        | STATUS  | NAME  |        PEER ADDRS        |       CLIENT ADDRS       | IS LEARNER |
+------------------+---------+-------+--------------------------+--------------------------+------------+
| 802d78549985d5a8 | started | k8scp | https://10.128.0.15:2380 | https://10.128.0.15:2379 |      false |
+------------------+---------+-------+--------------------------+--------------------------+------------+
```

6. Now that we know how many etcd databases are in the cluster, and their health, we can back it up. Use the `snapshot` argument to save the snapshot into the container data directory `/var/lib/etcd/`

```
student@cp:~$ kubectl -n kube-system exec -it etcd-k8scp -- sh -c "ETCDCTL_API=3 \
 ETCDCTL_CACERT=/etc/kubernetes/pki/etcd/ca.crt ETCDCTL_CERT=/etc/kubernetes/pki/etcd/server.crt \
 ETCDCTL_KEY=/etc/kubernetes/pki/etcd/server.key  etcdctl --endpoints=https://127.0.0.1:2379 \
 snapshot save /var/lib/etcd/snapshot.db "
```

```
{"level":"info","ts":1598380941.6584022,"caller":"snapshot/v3_snapshot.go:110","
msg":"created temporary db file","path":"/var/lib/etcd/snapshot.db.part"}
{"level":"warn","ts":"2020-08-25T18:42:21.671Z","caller":"clientv3/retry_interceptor.go
:116","msg":"retry stream intercept"}
{"level":"info","ts":1598380941.6736135,"caller":"snapshot/v3_snapshot.go:121","
msg":"fetching snapshot","endpoint":"https://127.0.0.1:2379"}
{"level":"info","ts":1598380941.7519674,"caller":"snapshot/v3_snapshot.go:134","
msg":"fetched snapshot","endpoint":"https://127.0.0.1:2379","took":0.093466104}
{"level":"info","ts":1598380941.7521122,"caller":"snapshot/v3_snapshot.go:143","
msg":"saved","path":"/var/lib/etcd/snapshot.db"}
Snapshot saved at /var/lib/etcd/snapshot.db
```

7. Verify the snapshot exists from the node perspective, the file date should have been moments earlier.

```
student@cp:~$ sudo ls -l /var/lib/etcd/
```

```
total 3888
drwx------ 4 root root    4096 Aug 25 11:22 member
-rw------- 1 root root 3973152 Aug 25 18:42 snapshot.db
```

8. Backup the snapshot as well as other information used to create the cluster both locally as well as another system in case the node becomes unavailable. Remember to create snapshots on a regular basis, perhaps using a cronjob to ensure a timely restore. When using the `snapshot restore` it's important the database not be in use. An HA cluster would remove and replace the control plane node, and not need a restore. More on the restore process can be found here: https://kubernetes.io/docs/tasks/administer-cluster/configure-upgrade-etcd/#restoring-an-etcd-cluster

```
student@cp:~$ mkdir $HOME/backup
student@cp:~$ sudo cp /var/lib/etcd/snapshot.db $HOME/backup/snapshot.db-$(date +%m-%d-%y)
student@cp:~$ sudo cp /root/kubeadm-config.yaml $HOME/backup/
student@cp:~$ sudo cp -r /etc/kubernetes/pki/etcd $HOME/backup/
```

Upgrade the Cluster

1. Begin by updating the package metadata for **APT**.

```
student@cp:~$ sudo apt update
```

```
Hit:1 http://us-central1.gce.archive.ubuntu.com/ubuntu bionic InRelease
Get:2 http://us-central1.gce.archive.ubuntu.com/ubuntu bionic-updates InRelease [88.7 kB]
Get:3 http://us-central1.gce.archive.ubuntu.com/ubuntu bionic-backports InRelease [74.6 kB]
Get:5 http://security.ubuntu.com/ubuntu bionic-security InRelease [88.7 kB]
<output_omitted>
```

2. View the available packages. The list will be long, you may have to scroll back up to the top to find a recent version. We will choose the `1.22.1` version.

```
student@cp:~$ sudo apt-cache madison kubeadm
```

```
   kubeadm |  1.22.2-00 | http://apt.kubernetes.io kubernetes-xenial/main amd64 Packages
   kubeadm |  1.22.1-00 | http://apt.kubernetes.io kubernetes-xenial/main amd64 Packages
   kubeadm |  1.22.0-00 | http://apt.kubernetes.io kubernetes-xenial/main amd64 Packages
   kubeadm |  1.21.5-00 | http://apt.kubernetes.io kubernetes-xenial/main amd64 Packages
   kubeadm |  1.21.4-00 | http://apt.kubernetes.io kubernetes-xenial/main amd64 Packages
   kubeadm |  1.21.3-00 | http://apt.kubernetes.io kubernetes-xenial/main amd64 Packages
   kubeadm |  1.21.2-00 | http://apt.kubernetes.io kubernetes-xenial/main amd64 Packages
<output_omitted>
```

3. Remove the hold on **kubeadm** and update the package.

```
student@cp:~$ sudo apt-mark unhold kubeadm
```

```
Canceled hold on kubeadm.
```

```
student@cp:~$ sudo apt-get install -y kubeadm=1.22.1-00
```

```
Reading package lists... Done
Building dependency tree
Reading state information... Done
<output_omitted>
```

4. Hold the package again to prevent updates along with other software.

```
student@cp:~$ sudo apt-mark hold kubeadm
```

```
kubeadm set on hold.
```

5. Verify the version of **Kubeadm** installed.

```
student@cp:˜$ sudo kubeadm version
```

```
kubeadm version: &version.Info{Major:"1", Minor:"22", GitVersion:"v1.22.1",
GitCommit:"632ed300f2c34f6d6d15ca4cef3d3c7073412212", GitTreeState:"clean",
BuildDate:"2021-08-19T15:44:22Z", GoVersion:"go1.16.7", Compiler:"gc",
Platform:"linux/amd64"}
```

6. To prepare the cp node for update we first need to evict as many pods as possible. The nature of daemonsets is to have them on every node, and some such as `Calico` must remain. Change the name to your node name, and ignore the daemonsets.

```
student@cp:˜$ kubectl drain k8scp --ignore-daemonsets
```

```
node/k8scp cordoned
WARNING: ignoring DaemonSet-managed Pods: kube-system/calico-node-r6rkh, kube-system/kube-proxy-kngq6
evicting pod kube-system/calico-kube-controllers-5447dc9cbf-6d7nw
evicting pod kube-system/coredns-66bff467f8-brl45
evicting pod kube-system/coredns-66bff467f8-q5ms8
pod/calico-kube-controllers-5447dc9cbf-6d7nw evicted
pod/coredns-66bff467f8-brl45 evicted
pod/coredns-66bff467f8-q5ms8 evicted
node/k8scp evicted
```

7. Use the **upgrade plan** argument to check the existing cluster and then update the software. You may notice that there are versions available later than v1.22.1. Use the v1.22.1 version. Read through the output and get a feel for what would be changed in an upgrade.

```
student@cp:˜$ sudo kubeadm upgrade plan
```

```
[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....
[preflight] Running pre-flight checks.
[upgrade] Running cluster health checks
[upgrade] Fetching available versions to upgrade to
[upgrade/versions] Cluster version: v1.21.2
[upgrade/versions] kubeadm version: v1.22.1
[upgrade/versions] Target version: v1.22.2
[upgrade/versions] Latest version in the v1.21 series: v1.21.5
<output_omitted>
```

8. We are now ready to actually upgrade the software. There will be a lot of output. Be aware the command will ask if you want to proceed with the upgrade, answer `yes`. Take a moment and look for any errors or suggestions, such as upgrading the version of etcd, or some other package.

```
student@cp:˜$ sudo kubeadm upgrade apply v1.22.1
```

```
[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 -o yaml'
[preflight] Running pre-flight checks.
[upgrade] Running cluster health checks
[upgrade/version] You have chosen to change the cluster version to "v1.22.1"
[upgrade/versions] Cluster version: v1.21.1
[upgrade/versions] kubeadm version: v1.22.1
[upgrade/confirm] Are you sure you want to proceed with the upgrade? [y/N]: y   #<--- Answer with y

<output_omitted>

[upgrade/successful] SUCCESS! Your cluster was upgraded to "v1.22.1". Enjoy!

```

```
[upgrade/kubelet] Now that your control plane is upgraded, please proceed with
upgrading your kubelets if you haven't already done so.
```

9. Check projectcalico.org or other CNI for supported version to match any updates. While the apply command may show some information projects which are not directly connected to Kubernetes may need to be updated to work with the new version.

10. Check the status of the nodes. The cp should show scheduling disabled. Also as we have not updated all the software and restarted the daemons it will show the previous version.

```
student@cp:~$ kubectl get node
```

```
NAME      STATUS                     ROLES                  AGE     VERSION
k8scp     Ready,SchedulingDisabled   control-plane,master   7h48m   v1.20.1
worker    Ready                      <none>                 7h46m   v1.20.1
```

11. Release the hold on **kubelet** and **kubectl**.

```
student@cp:~$ sudo apt-mark unhold kubelet kubectl
```

```
Canceled hold on kubelet.
Canceled hold on kubectl.
```

12. Upgrade both packages to the same version as **kubeadm**.

```
student@cp:~$ sudo apt-get install -y kubelet=1.22.1-00 kubectl=1.22.1-00
```

```
Reading package lists... Done
Building dependency tree
Reading state information... Done

<output_omitted>

Setting up kubelet (1.22.1-00) ...
Setting up kubectl (1.22.1-00) ...
```

13. Again add the hold so other updates don't update the Kubernetes software.

```
student@cp:~$ sudo apt-mark hold kubelet kubectl
```

```
kubelet set on hold.
kubectl set on hold.
```

14. Restart the daemons.

```
student@cp:~$ sudo systemctl daemon-reload

student@cp:~$ sudo systemctl restart kubelet
```

15. Verify the cp node has been updated to the new version. Then update other cp nodes using the same process except **sudo kubeadm upgrade node** instead of **sudo kubeadm upgrade apply**.

```
student@cp:~$ kubectl get node
```

```
NAME      STATUS                     ROLES                   AGE     VERSION
k8scp     Ready,SchedulingDisabled   control-plane,mastere   7h50m   v1.22.1
worker    Ready                      <none>                  7h48m   v1.21.1
```

16. Now make the cp available for the scheduler, again change the name to match the cluster node name on your control plane.

```
student@cp:~$ kubectl uncordon k8scp
```

```
node/k8scp uncordoned
```

17. Verify the cp now shows a `Ready` status.

```
student@cp:~$ kubectl get node
```

```
NAME       STATUS      ROLES                   AGE    VERSION
k8scp      Ready       control-plane,master    8h     v1.22.1
worker     Ready       <none>                  8h     v1.21.1
```

18. Now update the worker node(s) of the cluster. **Open a second terminal session to the worker**. Note that you will need to run a couple commands on the cp as well, having two sessions open may be helpful. Begin by allowing the software to update on the worker.

```
student@worker:~$ sudo apt-mark unhold kubeadm
```

```
Canceled hold on kubeadm.
```

19. Update the **kubeadm** package to the same version as the cp node.

```
student@worker:~$ sudo apt-get update && sudo apt-get install -y kubeadm=1.22.1-00
```

```
Hit:1 http://us-central1.gce.archive.ubuntu.com/ubuntu bionic InRelease

<output_omitted>

Setting up kubeadm (1.22.1-00) ...
```

20. Hold the package again.

```
student@worker:~$ sudo apt-mark hold kubeadm
```

```
kubeadm set on hold.
```

21. Back on the **cp terminal session** drain the worker node, but allow the daemonsets to remain.

```
student@cp:~$ kubectl drain worker --ignore-daemonsets
```

```
node/worker cordoned
WARNING: ignoring DaemonSet-managed Pods: kube-system/calico-node-nwm8w, kube-system/kube-proxy-66sh2
evicting pod default/before-688b465898-2sdx7
evicting pod default/after-6967f9db5-ws852
evicting pod kube-system/calico-kube-controllers-5447dc9cbf-5j947
<output_omitted>

pod/coredns-66bff467f8-nxclf evicted
pod/calico-kube-controllers-5447dc9cbf-5j947 evicted
node/worker evicted
```

22. Return to the **worker** node and download the updated node configuration.

```
student@worker:~$ sudo kubeadm upgrade node
```

```
[upgrade] Reading configuration from the cluster...
[upgrade] FYI: You can look at this config file with 'kubectl -n kube-system get cm
kubeadm-config -o yaml'
[preflight] Running pre-flight checks
[preflight] Skipping prepull. Not a control plane node.
[upgrade] Skipping phase. Not a control plane node.
[kubelet-start] Writing kubelet configuration to file "/var/lib/kubelet/config.yaml"
[upgrade] The configuration for this node was successfully updated!
[upgrade] Now you should go ahead and upgrade the kubelet package using your package manager.
```

23. Remove the hold on the software then update to the same version as set on the cp.

```
student@worker:~$ sudo apt-mark unhold kubelet kubectl
```

```
Canceled hold on kubelet.
Canceled hold on kubectl.
```

```
student@worker:~$ sudo apt-get install -y kubelet=1.22.1-00 kubectl=1.22.1-00
```

```
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following packages were automatically installed and are no longer required:

<output_omitted>

Preparing to unpack .../kubectl_1.20.1-00_amd64.deb ...
Unpacking kubectl (1.20.1-00) over (1.19.1-00) ...
Preparing to unpack .../kubelet_1.20.1-00_amd64.deb ...
Unpacking kubelet (1.20.1-00) over (1.19.1-00) ...
Setting up kubelet (1.20.1-00) ...
Setting up kubectl (1.20.1-00) ...
```

24. Ensure the packages don't get updated when along with regular updates.

```
student@worker:~$ sudo apt-mark hold kubelet kubectl
```

```
kubelet set on hold.
kubectl set on hold.
```

25. Restart daemon processes for the software to take effect.

```
student@worker:~$ sudo systemctl daemon-reload

student@worker:~$ sudo systemctl restart kubelet
```

26. Return to the cp node. View the status of the nodes. Notice the worker status.

```
student@cp:~$  kubectl get node
```

```
NAME      STATUS                      ROLES                  AGE   VERSION
k8scp     Ready                       control-plane,master   8h    v1.22.1
worker    Ready,SchedulingDisabled    <none>                 8h    v1.22.1
```

27. Allow pods to be deployed to the worker node.

```
student@cp:~$ kubectl uncordon worker
```

```
node/worker uncordoned
```

28. Verify the nodes both show a `Ready` status and the same upgraded version.

```
student@cp:~$ kubectl get nodes
```

```
NAME     STATUS   ROLES                  AGE   VERSION
k8scp    Ready    control-plane,master   8h    v1.22.1
worker   Ready    <none>                  8h   v1.22.1
```