[Compute] Managing State With Deployments, Scheduling

Pod / Replicaset 기본 실습

- Simple Pod 생성

  # kubectl run test --image=nginx

- 생성한 pod 확인

  # kubectl get pod test

  # kubectl describe pod test

- Replicaset 확인/생성

  # kubeclt get replicasets

  # kubectl create replicaset test ~

- Pod/Replicaset 삭제

  # kubectl delete pod / replicaset ~

- Replicaset을 edit 하여 pod 숫자 확인

  # kubectl edit replicaset ~

꾸준한 연습만이 합격의 지름길

익숙해져야 합니다

Deployment 기본 실습

- Deployment 생성 및 수정

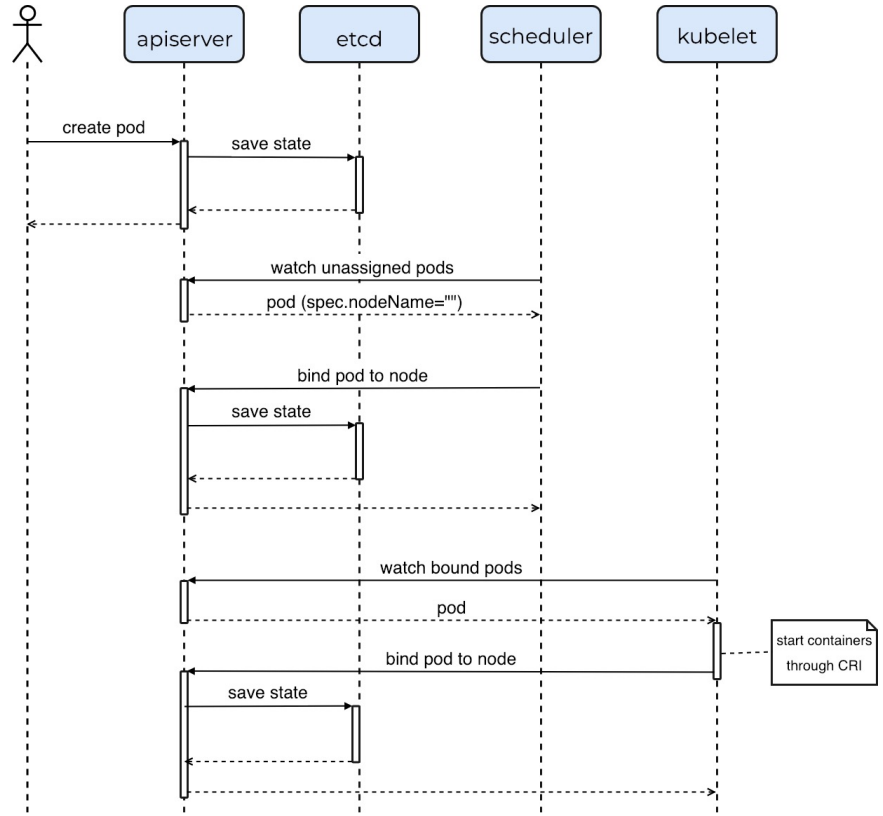
```
[centos@osk-master-01 ~]$ kubectl create deployment nginx_deployment --image=nginx --dry-run=client -o yaml > deployment.yaml
[centos@osk-master-01 ~]$ vi deployment.yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  creationTimestamp: null
  labels:
    app: nginx_deployment
  name: nginx_deployment
spec:
  replicas: 1
  selector:
    matchLabels:
      app: nginx_deployment
  strategy: {}
  template:
    metadata:
      creationTimestamp: null
      labels:
        app: nginx_deployment
    spec:
      containers:
      - image: nginx
        name: nginx
        resources: {}
status: {}
```

꾸준한 연습만이 합격의 지름길

익숙해져야 합니다

Scheduler

- 컨테이너를 적절한 노드에 스케줄링(배치) 하는 역할(전용 바이너리)



- 스케쥴러는 nodeName이 없는 파드가 있는지 API 서버를 지속 감시
- 해당 파드를 적합한 노드로 선택하고, 파드의 nodeName 업데이트
- 해당 노드의 쿠블렛이 계속 API서버를 보다가 자기꺼면 실행

그림 출처 : https://banzaicloud.com/blog/k8s-custom-scheduler/

Scheduling 프로세스

- 스케줄러가 파드를 배치하는 주요 기준

  1) 사전 조건

  - 파드가 특정 노드에 적합한지 확인.
  - ex) 파드에서 요청한 메모리 양을 노드가 감당할 수 있을지, 유저가 node selector로 지정한 경우 등

  2) 우선순위

  - 파드가 노드에서 실행할 수 있다고 가정 후, 상대적 가치를 판단
  - ex) 이미지가 이미 존재하는 노드에 가중치를 부여 : 빠른 시작 가능, 동일한 Service 내 있는 파드라면 spreading 시켜놓음

※ 스케줄링은 한 순간에만 최적이므로, 이후 충돌 등 여러 이유로 실행중인 파드를 이동할 수 도 있음. 이 경우 해당 pod를 삭제하고 재 생성

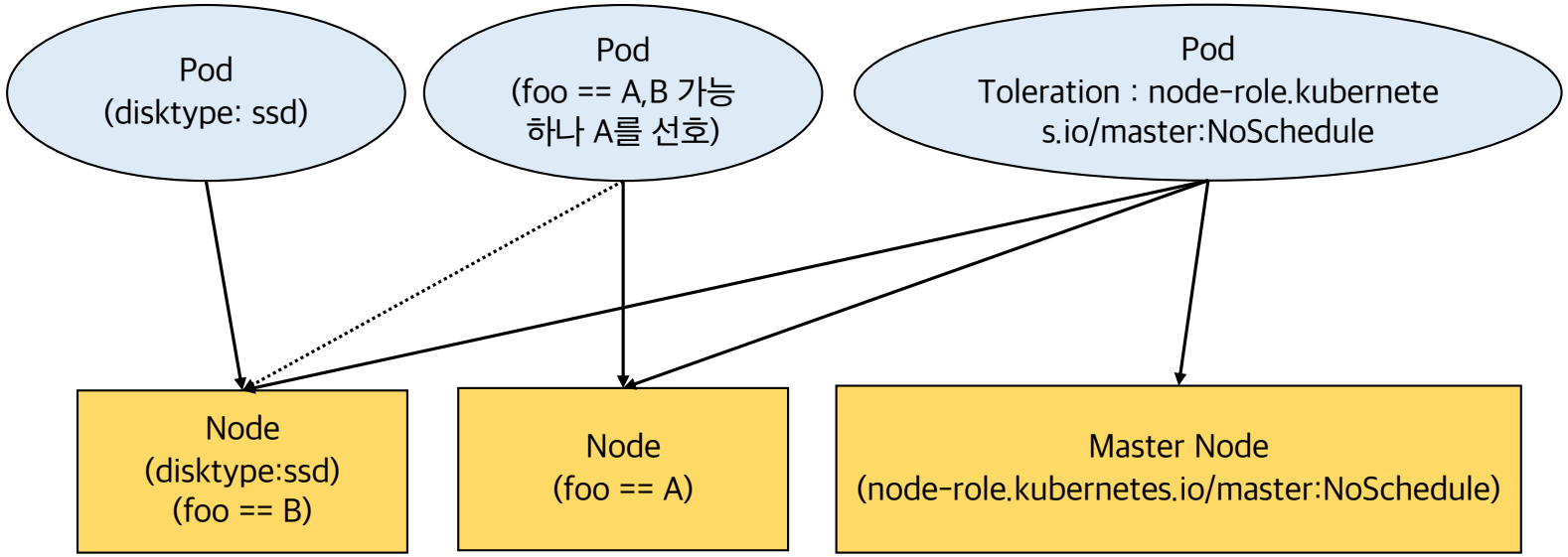
Policy 정책

- 쿠버네티스 스케쥴링과 다르게 운용자가 직접 스케쥴 하고 싶다면 주로 아래 3가지 방안을 활용

  1) Node Selector

  2) Node Affinity

  3) Taint/Toleration



Node Selector

- Node Selector

- 노드에도 파드에도 레이블을 부여하여 동일한 조건에 맞게 스케쥴 되도록 설정

https://kubernetes.io/ko/docs/concepts/scheduling-eviction/assign-pod-node/

(1) 노드에 레이블 붙이기. (예) `#kubectl label nodes kubernetes-foo-node-1.c.a-robinson.internal disktype=ssd`

(2) 파드에 node selector 추가하기

```
spec:
  containers:
  nodeSelector:
    disktype: ssd
```

Node Affinity

- Node Selector는 선택에 대한 요구사항을 이야기하는 사전 조건
- 복잡한 선택에는 Affinity가 유연. (ex) 레이블 foo이면 A 또는 B 노드를 선택해주세요.

```
kind: Pod
...
spec:
  affinity:
    nodeAffinity:
      requiredDuringSchedulingIgnoredDuringExecution:
        nodeSelectorTerms:
        - matchExpressions:
          # foo == A or B
          - key: foo
            operator: In
            values:
            - A
            - B
 ...
```

Node Affinity

- Node Affinity

  - (ex) A, B 둘다 가능하지만, A로 라벨링 된 노드를 더 선호합니다.

```
kind: Pod
...
spec:
  affinity:
    nodeAffinity:
      requiredDuringSchedulingIgnoredDuringExecution:
       nodeSelectorTerms:
       - matchExpressions:
         # foo == A or B
         - key: foo
           operator: In
           values:
           - A
           - B
      preferredDuringSchedulingIgnoredDuringExecution:
       preference:
       - weight: 1
         matchExpressions:
         # foo == A
         - key: foo
           operator: In
           values:
           - A
...
```

[operator]

In : 해당 Value가 있는지?(키+쌍 모두)

Exists : Key만 있으면 됨(키만 확인)

Taint & Toleration

- Taint : 스케쥴러가 pod를 배치할 때 Taint (오염)되어 있는 node들은 배치하지 않도록 함
- Toleration : Taint가 묻어 있어도 배치할 수 있도록 함

- Kubeadm은 설치 시, 자동으로 Controler Plane에 node-role.Kubernetes.io/master 테인트로 제어하여 일반 파드와 분리

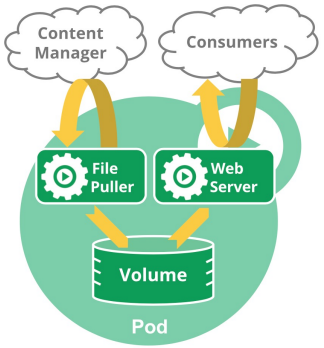
```
[root@osk-master-01 pki]# kubectl describe node osk-master-01
Name:               osk-master-01.kr-central-1.c.internal
Roles:              control-plane,master
Labels:             beta.kubernetes.io/arch=amd64
                    beta.kubernetes.io/os=linux
                    kubernetes.io/arch=amd64
                    kubernetes.io/hostname=osk-master-01.kr-central-1.c.internal
                    kubernetes.io/os=linux
                    node-role.kubernetes.io/control-plane=
                    node-role.kubernetes.io/master=
                    node.kubernetes.io/exclude-from-external-load-balancers=
Annotations:        kubeadm.alpha.kubernetes.io/cri-socket: /var/run/dockershim.sock
                    node.alpha.kubernetes.io/ttl: 0
                    projectcalico.org/IPv4Address: 172.30.5.193/22
                    projectcalico.org/IPv4IPIPTunnelAddr: 192.168.205.192
                    volumes.kubernetes.io/controller-managed-attach-detach: true
CreationTimestamp:  Sat, 14 Aug 2021 11:30:14 +0000
Taints:             node-role.kubernetes.io/master:NoSchedule
Unschedulable:      false
```

멀티 컨테이너 Pod

- Deployment 를 파일로 수정
- (실수 주의) Container가 2번 들어가는 것이 아님

```
spec:
  containers:
  - image: busybox
    name: lemon
  containers:
  - image: redis
    name: gold
```

- 두개의 컨테이너가 한개의 emptydir 볼륨을 바라보게 하기
  https://kubernetes.io/docs/concepts/storage/volumes/#emptydir-configuration-example




```yaml
apiVersion: v1
kind: Pod
metadata:
 name: test-pd
spec:
 containers:
 - image: k8s.gcr.io/test-webserver
   name: test-container
   volumeMounts:
   - mountPath: /cache
     name: cache-volume
 - image: k8s.gcr.io/test-webserver
   name: side-car
   volumeMounts:
   - mountPath: /cache
     name: cache-volume
 volumes:
 - name: cache-volume
   emptyDir: {}
```

Namespace

- 쿠버네티스는 동일한 물리 클러스터를 기반으로 하는 여러 가상 클러스터를 지원, 이런 가상 클러스터가 네임스페이스
- 네임스페이스 확인

```
[centos@osk-master-01 ~]$ kubectl get pods -A
NAMESPACE     NAME                                       READY   STATUS    RESTARTS   AGE
kube-system   calico-kube-controllers-58497c65d5-wnfnv   1/1     Running   0          22h
```

- 네임스페이스 생성, 해당 네임스페이스에 파드 생성

```
[centos@osk-master-01 ~]$ kubectl create namespace kakao
namespace/kakao created
[centos@osk-master-01 ~]$ kubectl run likelion --image=nginx --namespace=kakao
pod/likelion created
```

- 생성한 파드 확인

```
[centos@osk-master-01 ~]$ kubectl get pods
No resources found in default namespace.
[centos@osk-master-01 ~]$ kubectl get pods -n kakao
NAME       READY   STATUS    RESTARTS   AGE
likelion   1/1     Running   0          20s
```

Label과 Selector

- [centos@osk-master-01 ~]$ kubectl get pod --help
 Display one or many resources
   Prints a table of the most important information about the specified resources. You can filter the list using a label selector and t
he --selector flag. (생략)

```
[centos@osk-master-01 ~]$ kubectl run pod --image=nginx -l teacher=osk --dry-run=client -o yaml > label.yaml
[centos@osk-master-01 ~]$ cat label.yaml
apiVersion: v1
kind: Pod
metadata:
  creationTimestamp: null
  labels:
    teacher: osk
  name: pod
spec:
  containers:
  - image: nginx
    name: pod
    resources: {}
  dnsPolicy: ClusterFirst
  restartPolicy: Always
status: {}
[centos@osk-master-01 ~]$ kubectl apply -f label.yaml
pod/pod created
[centos@osk-master-01 ~]$ kubectl get pod --selector teacher=osk
NAME   READY   STATUS    RESTARTS   AGE
pod    1/1     Running   0          23s
[centos@osk-master-01 ~]$
[centos@osk-master-01 ~]$ kubectl get all --selector teacher=osk
NAME      READY   STATUS    RESTARTS   AGE
pod/pod   1/1     Running   0          39s
[centos@osk-master-01 ~]$
```

Staticpod

- Static pod(Manifest)는 kubectl 및 control plane에서 관리하지 않는 pod
- pod spec을 디스크에 직접 쓸수 있으며, kubelet은 시작과 함께 바로 지정된 컨테이너를 시작.
- Kubelet은 지속적으로 변경사항이 있는지 Manifest 파일을 지속적으로 모니터링

```
[root@osk-master-01 manifests]# pwd
/etc/kubernetes/manifests
[root@osk-master-01 manifests]#
[root@osk-master-01 manifests]# ls
etcd.yaml  kube-apiserver.yaml  kube-controller-manager.yaml  kube-scheduler.yaml
[root@osk-master-01 manifests]#
[root@osk-master-01 manifests]# kubectl get pod -A -o wide
NAMESPACE     NAME                                                       READY   STATUS    RESTARTS   AGE   IP               NODE                                   NOMINATED NODE   READINESS GATES
kube-system   calico-kube-controllers-58497c65d5-wnfnv                   1/1     Running   0          14h   192.168.205.193  osk-master-01.kr-central-1.c.internal  <none>           <none>
kube-system   calico-node-9nc4s                                          1/1     Running   0          14h   172.30.7.246     osk-master-03.kr-central-1.c.internal  <none>           <none>
kube-system   calico-node-b4ghk                                          1/1     Running   0          14h   172.30.5.179     osk-worker-02.kr-central-1.c.internal  <none>           <none>
kube-system   calico-node-dqx6d                                          1/1     Running   0          14h   172.30.5.3       osk-worker-01.kr-central-1.c.internal  <none>           <none>
kube-system   calico-node-gg6vc                                          1/1     Running   0          14h   172.30.5.193     osk-master-01.kr-central-1.c.internal  <none>           <none>
kube-system   calico-node-mvbn5                                          1/1     Running   0          14h   172.30.4.105     osk-master-02.kr-central-1.c.internal  <none>           <none>
kube-system   coredns-558bd4d5db-ctf7n                                   1/1     Running   0          14h   192.168.205.194  osk-master-01.kr-central-1.c.internal  <none>           <none>
kube-system   coredns-558bd4d5db-q8zph                                   1/1     Running   0          14h   192.168.205.195  osk-master-01.kr-central-1.c.internal  <none>           <none>
kube-system   etcd-osk-master-01.kr-central-1.c.internal                 1/1     Running   0          14h   172.30.5.193     osk-master-01.kr-central-1.c.internal  <none>           <none>
kube-system   etcd-osk-master-02.kr-central-1.c.internal                 1/1     Running   0          14h   172.30.4.105     osk-master-02.kr-central-1.c.internal  <none>           <none>
kube-system   etcd-osk-master-03.kr-central-1.c.internal                 1/1     Running   0          14h   172.30.7.246     osk-master-03.kr-central-1.c.internal  <none>           <none>
kube-system   kube-apiserver-osk-master-01.kr-central-1.c.internal       1/1     Running   0          14h   172.30.5.193     osk-master-01.kr-central-1.c.internal  <none>           <none>
kube-system   kube-apiserver-osk-master-02.kr-central-1.c.internal       1/1     Running   0          14h   172.30.4.105     osk-master-02.kr-central-1.c.internal  <none>           <none>
kube-system   kube-apiserver-osk-master-03.kr-central-1.c.internal       1/1     Running   0          14h   172.30.7.246     osk-master-03.kr-central-1.c.internal  <none>           <none>
kube-system   kube-controller-manager-osk-master-01.kr-central-1.c.internal   1/1     Running   1          14h   172.30.5.193     osk-master-01.kr-central-1.c.internal  <none>           <none>
kube-system   kube-controller-manager-osk-master-02.kr-central-1.c.internal   1/1     Running   0          14h   172.30.4.105     osk-master-02.kr-central-1.c.internal  <none>           <none>
kube-system   kube-controller-manager-osk-master-03.kr-central-1.c.internal   1/1     Running   0          14h   172.30.7.246     osk-master-03.kr-central-1.c.internal  <none>           <none>
kube-system   kube-proxy-h4hqp                                           1/1     Running   0          14h   172.30.5.3       osk-worker-01.kr-central-1.c.internal  <none>           <none>
kube-system   kube-proxy-jvnsv                                           1/1     Running   0          14h   172.30.5.193     osk-master-01.kr-central-1.c.internal  <none>           <none>
kube-system   kube-proxy-qb95g                                           1/1     Running   0          14h   172.30.7.246     osk-master-03.kr-central-1.c.internal  <none>           <none>
kube-system   kube-proxy-s7kl6                                           1/1     Running   0          14h   172.30.5.179     osk-worker-02.kr-central-1.c.internal  <none>           <none>
kube-system   kube-proxy-tkzjh                                           1/1     Running   0          14h   172.30.4.105     osk-master-02.kr-central-1.c.internal  <none>           <none>
kube-system   kube-scheduler-osk-master-01.kr-central-1.c.internal       1/1     Running   1          14h   172.30.5.193     osk-master-01.kr-central-1.c.internal  <none>           <none>
kube-system   kube-scheduler-osk-master-02.kr-central-1.c.internal       1/1     Running   0          14h   172.30.4.105     osk-master-02.kr-central-1.c.internal  <none>           <none>
kube-system   kube-scheduler-osk-master-03.kr-central-1.c.internal       1/1     Running   0          14h   172.30.7.246     osk-master-03.kr-central-1.c.internal  <none>           <none>
[root@osk-master-01 manifests]#
```

Staticpod

- 기본 값 외우지 않고 static pod manifest 저장된 위치 찾기

```
[centos@osk-master-01 ~]$ ps -ef ¦ grep kubelet ¦ grep conf
root     13731     1  3  8월14 ?      00:46:03 /usr/bin/kubelet --bootstrap-kubeconfig=/etc/kubernetes/bootstrap
-kubelet.conf --kubeconfig=/etc/kubernetes/kubelet.conf --config=/var/lib/kubelet/config.yaml --network-plugin=c
ni --pod-infra-container-image=k8s.gcr.io/pause:3.4.1


[centos@osk-master-01 ~]$ sudo cat /var/lib/kubelet/config.yaml
(생략)
shutdownGracePeriod: 0s
shutdownGracePeriodCriticalPods: 0s
staticPodPath: /etc/kubernetes/manifests
streamingConnectionIdleTimeout: 0s
syncFrequency: 0s
volumeStatsAggPeriod: 0s
```

Staticpod

- Static pod 만들기

```
[centos@osk-master-01 ~]$ kubectl run --restart=Never --image=busybox static-busybox --dry-run=client -o yaml --
command -- sleep 1000 > ./static-busybox.yaml

[centos@osk-master-01 ~]$ cat static-busybox.yaml
apiVersion: v1
kind: Pod
metadata:
  creationTimestamp: null
  labels:
    run: static-busybox
  name: static-busybox
spec:
  containers:
  - command:
    - sleep
    - "1000"
    image: busybox
    name: static-busybox
    resources: {}
  dnsPolicy: ClusterFirst
  restartPolicy: Never
status: {}
```

클러스터 노드 작업

- Cordon/Drain/Taint 비교

cor·don

1. [군사] 초병선(哨兵線); 비상[경계]선 2. 방역선, 교통 차단선 3. 수장으로 장식하다

- 노드에 더이상 어떤 파드도 스케쥴링 되지 않도록 하는 기능
- ex) 오늘 야간에 정기점검이 잡혀있는 노드

---

drain

1. 물을 빼내다 2. (액체를) 따라 내다 3. 배수관

- 노드에 있는 파드를 지금 바로 다른 노드로 이동시키는 기능
- Drain 명령어 입력 시 Cordon을 수행 후, Drain 하게 됨
- ex) 해당 노드에 고장이 발생하여 파드를 다른 곳에서 새로 시작해야 하는 경우

---

taint

1. (평판 등을) 더럽히다, 오염시키다, 오점을 남기다 2. 오점, 오명

- 파드가 부적절한 노드에 실행되지 않도록 설정하는 기능
- ex) Toleration과 결합하여 특정 파드만 실행시키고 싶을 때 (Control Plane으로 쓰는 Master Node 등)

클러스터 업그레이드

- 참고 : https://kubernetes.io/docs/tasks/administer-cluster/kubeadm/kubeadm-upgrade/
- Kubelet과 kubeadm 자체는 kubeadm으로 관리되지 않기에, OS 패키지 관리자인 apt/yum으로 설치 필요

- 업그레이드 순서 : 컨트롤 플레인 1번 노드 먼저 업그레이드 ▶ 2/3번 마스터 업그레이드 ▶ 워커 노드 업그레이드 수행
  (참고1) 업그레이드 시, kubectl cordon/drain을 활용하여 사용자 워크로드를 조정토록 함
  (참고2) 업그레이드가 완료 된 후에는 uncordon 하여 다시 파드가 스케쥴링 되도록 설정

```
[1번 노드] #sudo yum list --showduplicates kubeadm --disableexcludes=Kubernetes
[1번 노드] #sudo yum install -y kubeadm-1.22.1-0 --disableexcludes=kubernetes
[1번 노드] #kubeadm upgrade plan
```

→ preflight 수행하여 클러스터가 정상인지 확인, kube-system kubeadm-config 컨피그맵 검사
→ 앞으로 업그레이드하는게 좋을지 알려줌 (root 계정으로 실행)

```
[1번 노드] #kubeadm upgrade apply v1.22.1
[1번 노드] #kubectl uncordon master01
```

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

클러스터 업그레이드

- 참고 : https://kubernetes.io/docs/tasks/administer-cluster/kubeadm/kubeadm-upgrade/
- Kubelet과 kubeadm 자체는 kubeadm으로 관리되지 않기에, OS 패키지 관리자인 apt/yum으로 설치 필요

- 업그레이드 순서 : 컨트롤 플레인 1번 노드 먼저 업그레이드 ▶ 2/3번 마스터 업그레이드 ▶ 워커 노드 업그레이드 수행

```
[2,3번 노드] #sudo yum install -y kubeadm-1.22.1-0 --disableexcludes=kubernetes
[2,3번 노드] #kubeadm upgrade node
[2,3번 노드] #kubectl uncordon master02
```

```
[upgrade] The control plane instance for this node was successfully updated!
[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.
```

```
[1~3번 노드] #sudo yum install -y kubelet-1.22.1-0 kubectl-1.22.1-0 --disableexcludes=kubernetes
[1~3번 노드] #sudo systemctl daemon-reload
[1~3번 노드] #sudo systemctl restart kubelet
[1~3번 노드] kubectl uncordon osk-master-01.kr-central-1.c.internal osk-master-02.kr-central-1.c.internal os
k-master-03.kr-central-1.c.internal
```

클러스터 업그레이드

- 참고 : https://kubernetes.io/docs/tasks/administer-cluster/kubeadm/kubeadm-upgrade/
- Kubelet과 kubeadm 자체는 kubeadm으로 관리되지 않기에, OS 패키지 관리자인 apt/yum으로 설치 필요

- 업그레이드 순서 : 컨트롤 플레인 1번 노드 먼저 업그레이드 ▶ 2/3번 마스터 업그레이드 ▶ 워커 노드 업그레이드 수행

```
[워커노드1,2] #sudo yum install -y kubeadm-1.22.1-0 --disableexcludes=Kubernetes
[워커노드1,2] #sudo kubeadm upgrade node
[centos@osk-master-01 ~]$ kubectl drain osk-worker-01.kr-central-1.c.internal --ignore-daemonsets
[워커노드1] #sudo yum install -y kubelet-1.22.1-0 kubectl-1.22.1-0 --disableexcludes=kubernetes
[워커노드1] #sudo systemctl daemon-reload
[워커노드1] #sudo systemctl restart kubelet
[centos@osk-master-01 ~]$ kubectl uncordon osk-worker-01.kr-central-1.c.internal
[centos@osk-master-01 ~]$ kubectl get nodes
NAME                                   STATUS   ROLES                  AGE    VERSION
osk-master-01.kr-central-1.c.internal   Ready    control-plane,master   127m   v1.22.1
osk-master-02.kr-central-1.c.internal   Ready    control-plane,master   121m   v1.22.1
osk-master-03.kr-central-1.c.internal   Ready    control-plane,master   116m   v1.22.1
osk-worker-01.kr-central-1.c.internal   Ready    <none>                 105m   v1.22.1
osk-worker-02.kr-central-1.c.internal   Ready    <none>                 105m   v1.21.0
[워커노드2도 drain부터 반복]
```