```
openssl genrsa -out dv.key 2048
openssl req -new -key dv.key -subj "/CN=admin/O=system:masters" -out admin.csr
openssl req -in csr.txt -noout -text
openssl x509 -req -in dv.csr -CA ca.crt -CAkey ca.key -out dv.cer
```

```
alias set='k config set-context `k config current-context` '
kubectl config set-context --current --namespace=deve
alias k=kubectl
complete -F __start_kubectl k
```

```
spec: NETWOTK POLICY INGRESS/EGRESS
  # podSelector: {} matches all pods in namespace
  podSelector:
    matchLabels:
# Applies to all the pods with role web
      role: web
  policyTypes:
  - Ingress
  - Egress
  ingress:
    # - {} allow all ingress
  - from:
    - podSelector:
        matchLabels:
          role: frontend
    ports:
    - protocol: TCP
      port: 80
  egress:
    # - {} allow all ingress
  - to:
    - podSelector:
        matchLabels:
          role: backend
    ports:
    - protocol: TCP
      port: 80
```

```
apiVersion: v1. Persistent Volume
kind: PersistentVolume
metadata:
  name: persistent-volumes
spec:
  capacity:
    storage: 1Gi
```

```
- image: nginx. ConfigMap  use
    name: nginxpod
    envFrom:
    - configMapRef:
        name: sample-configmap
```

```
resources: {} Securiy Context
    # Can be defined at spec level however container level will override. Also capabilities applicable only with in container
    securityContext:
      runAsUser: user1
      capabilities:
        add: [ "MAC_ADMIN", "SYS_TIME"]
```

```
apiVersion: v1. Persistent Volume
kind: PersistentVolume
metadata:
  name: persistent-volumes
spec:
  capacity:
    storage: 1Gi
  accessModes:
  - ReadWriteOnce
  hostPath:  # At same level are capacity
    path: /tmp


apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: claim-log-1
spec:
  accessModes:
    - ReadWriteOnce
  volumeMode: Filesystem
  resources:
    requests:
      storage: 50Mi
```

```
spec: Volumes
  containers:
  - image: dennysv/alpine-nginx-version1.0
    name: voleme-container
    volumeMounts:
    - mountPath: /cache
      name: cache-volume
    - mountPath: /tmpmount
      name: temp-volume
  volumes:
    # Creatng empty volume for sharing data with in pods. It will get destroyed when pod terminated
  - name: cache-volume
    emptyDir: {}
    # Creating volume from local Path
  - name: temp-volume
    hostPath:
      path: /tmp
      type: Directory
```

```
spec:  NodeAffinity
  containers:
  - image: nginx
    name: affinity
    resources: {}
  affinity:
      nodeAffinity:

requiredDuringSchedulingIgnoredDuringExecution:
          nodeSelectorTerms:
          - matchExpressions:
            - key: kubernetes.io/os
              operator: In
              values:
              - linux
              - amd64
```

```
    resources: {} Tolerations
    tolerations:
    - key: "env"
      value: "Prod"
      effect: "NoSchedule"
      operator: "Equal"
```

Upgrade

```
apt upgrade kubeadm=1.12.0-00
kubeadm upgrade apply v1.12.0
apt upgrade kubelet=1.12.0-00
systemctl restart kubelet
```

```
k drain node01 –ignore-daemonsets
upgrade kubeadm,kubelet
kubeadm upgrade node config –kubelet-version v1.12.0
systemctl restart kubeadm
uncordon
```

```
kubectl get pods -o=jsonpath='{.items[0].metadata.name}'
kubectl get pods -o=jsonpath="{.items[*]['metadata.name', 'status.capacity']}"
k get nodes -o jsonpath='{.items[*].status.addresses[?(@.type=="ExternalIP")].address}'
```