Best Practice of Kubernetes

K8s Training

Notes: if you are using M1, you can't use the virtual machine, fusion can't support the M1, so you should use the cloud host.

Prepare:

System: Centos [3]

Docker

1. Download the docker

```
cd /etc/yum.repos.d/
wget https://mirrors.aliyun.com/docker-ce/linux/centos/docker-ce.repo
yum install -y docker-ce
systemctl enable --now docker
```

1. try the docker command in the terminal

```
docker search   # search image from local
docker pull     # pull image from docker hub
docker push     # push image from docker hub
docker images   # list docker images
docker rmi      # remove the image
docker run      # run the image, -P will assign the port automation, -p will set the port use user set.
docker build      # Create image.

docker ps –a -l # list the container
docker stop     # stop the container
docker rm       # remove the container
docker start    # start a container
docker attach | exec
# attach: when the docker starts, it will create a new process in your computer, attach will use this process and into the container. This will affect the docker.
# exec: it will create a new process in your computer, and operate the container, if this process is killed, don't affect the container process.

docker save     # save the image as a tar file in the local
docker load     # load image from the tar file.
docker tag      # add a tag to the image
```

1. Dockerfile every command is one layout, if see the same layout in the other image, the docker pull won't pull it again. If you install some management tool, like yum, clearing the cache will decrease the image size.

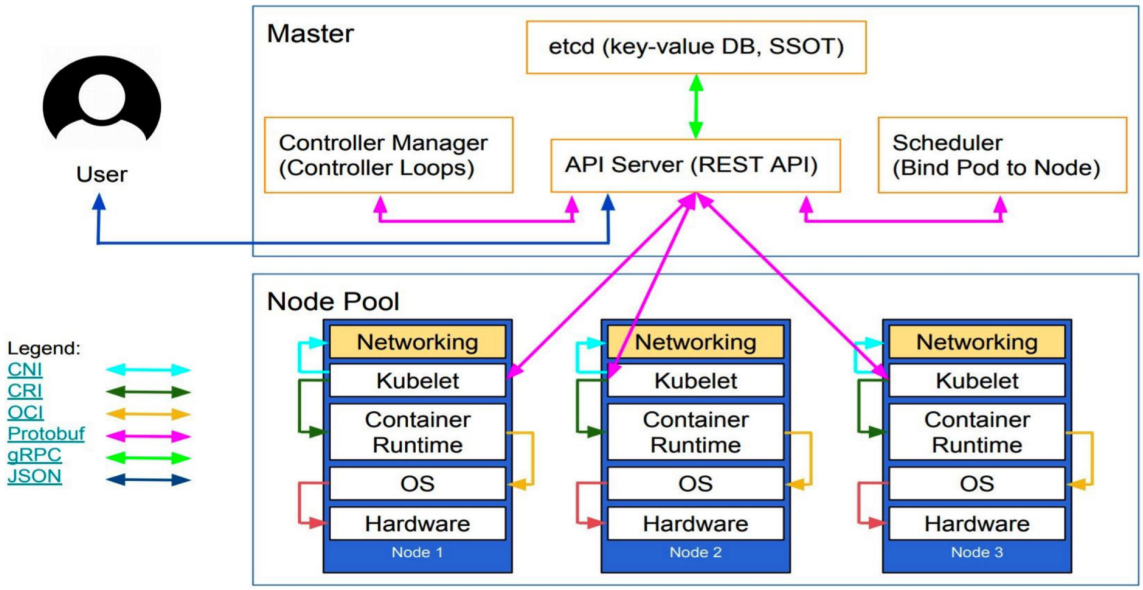
```
FROM      #  base image, if you want to do some system image, can't use this.

RUN       # run some commands in this command, like cp, mv, sed, clear cache

VOLUME    # config the storage

CMD       # some start command

EXPOSE    # expose the port that which user can visit from the outside.
```

K8s

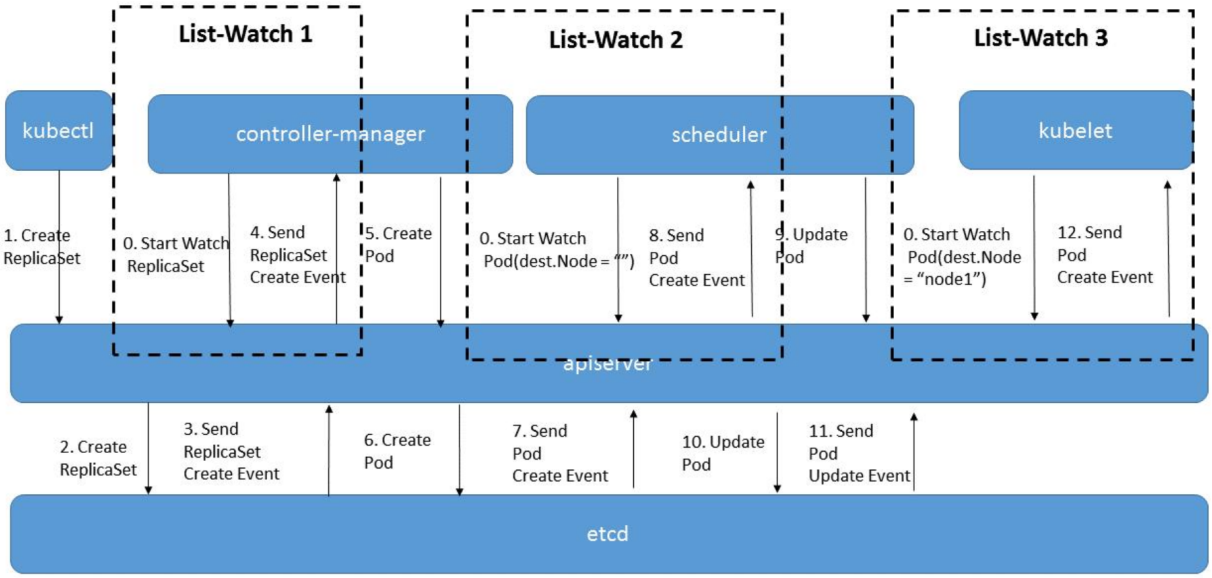
Prepare

1. git clone https://github.com/unixhot/salt-kubeadm.git
2. Config the env base on this

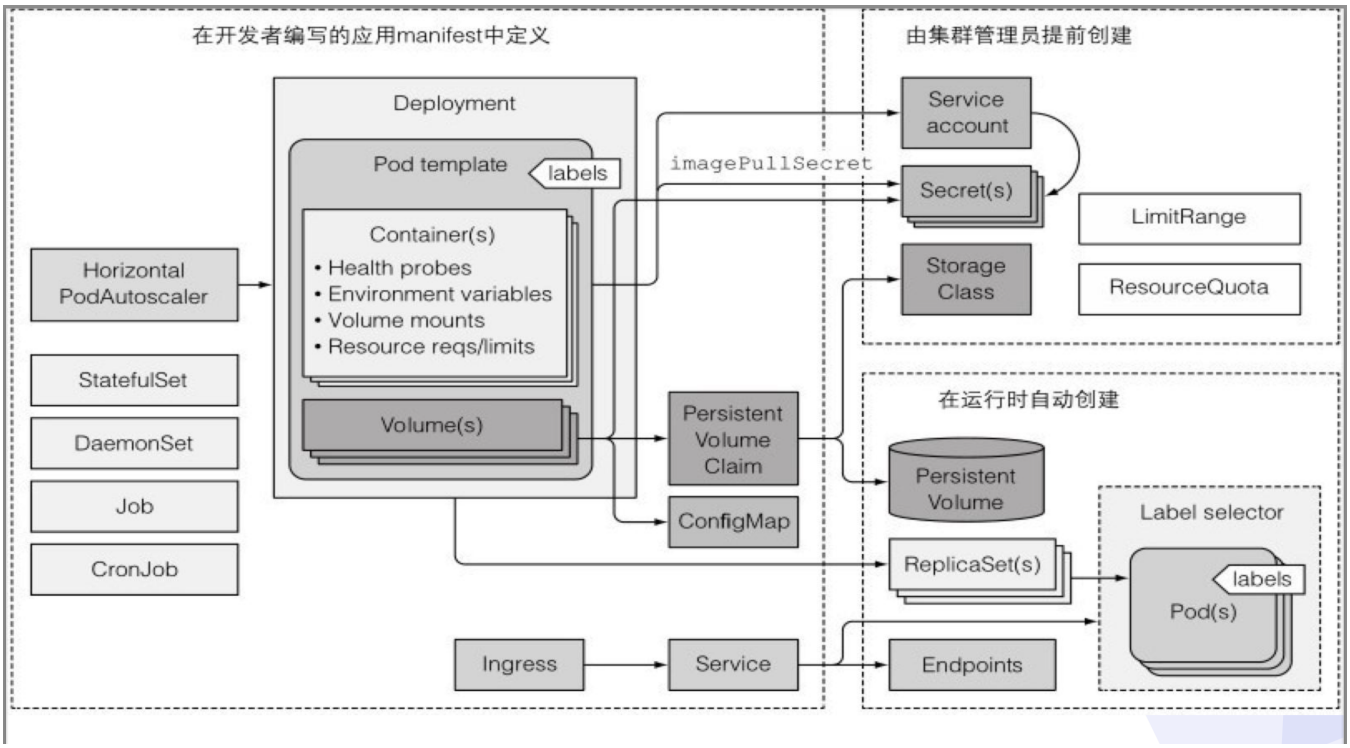
K8s Framework:



K8s use list-watch model, the user sends api to api resource, this api use `Transfer-Encoding: chunked` waiting, other resources watch the request and do something to create the resource. If any resource doesn't finish, the api will resend and wait.



All the resources:



Notes:

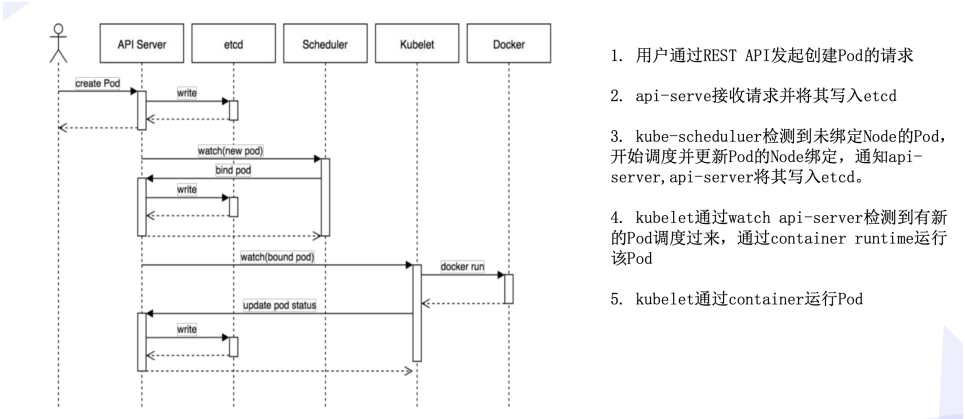
```
when you run the ***salt-ssh '*' state.highstate***  , please wait and don't break it, it will download kubeadm kubeletdocker and need a long time.
Before you run the kubeadm init, please change the name to your node1's hostname in the `/etc/sysconfig/kubeadm.yml`, if you don't do this, you need to run `kubeadm reset -y `, this will remove all the resource you create after kubeadm init. So, take care.
```

Config K8s Controls

1. **Pods**

   the pod is the minimum k8s unit, you can put multiple containers in one pod.

   Creat pod flow:



1. 用户通过REST API发起创建Pod的请求
2. api-serve接收请求并将其写入etcd
3. kube-scheduluer检测到未绑定Node的Pod，开始调度并更新Pod的Node绑定，通知api-server,api-server将其写入etcd。
4. kubelet通过watch api-server检测到有新的Pod调度过来，通过container runtime运行该Pod
5. kubelet通过container运行Pod

   Know how to write a pod base on a template

   Auto scale when the cpu is more than average. See HPA(horizontal pod autoscalers)

   ```
   kubectl get hpa
   ```

   Scheduler:

- No Scheduler
- Scheduler
- nodeSelector, nodeName
- NodeAffinity (node), PodAffinity (pod), PodAntiAffinity (pod).
  requiredDuringSchedulingIgnoredDuringExecution and preferredDuringSchedulingIgnoredDuringExecution.
  Operation: IN, NotIn, Exists, DoesNotExist
- Taints (node), Toleration (pod)

Pod health check:

- ExecAction # run exec to check
- TCPSocketAction # Send ip:port to check.
- HTTPGetAction # send request to service, like path call /info api

Pod exists check:

- Liveness/Readiness
  Liveness/Readiness can use the 3 methods above.

The check is in the kubelet

2. deployment

   know how to change the deploy yaml base on a template.

   Know how to roll back deploy.
   Notes: if revisionHistoryLimit is 0, you can't roll back, the default is 10.

   ```
   kubectl rollout
   ```

   Know how to change the deploy: edit, set.

   ```
   kubectl edit|set
   ```

3. Service

   Know how to change the service yaml base on a template

   Know how to set nodePorts.
4. Ingress

   For the ingress, we set the port as a service port, not the nodePort.
5. PV(Persistent Volumes)

   PV isn't in the namespace, if you set the path, please confirm you create the fold, PV doesn't create the folder automation.

   Pod can't operate the PV directly, it should use PVC link to a PV.

   Unit: 1Gi = 1024M 1G = 1000M
6. PVC(Persistent Volumes Claims)

   PVC is in the namespace, you can use the PVC link to the PV.
7. Node

   cordon: prevent the pod schedule this Node.

   ```
   kubectl cordon|uncordon node
   ```

   drain: clear up all the pods from this Node.

   ```
   kubectl drain node
   ```

   Schedule: See Node
8. ReplicaSet or ReplicaControl

   control the pod number, you can change it using this command:
   `kubectl scale --replicas=n rsname`
9. Endpoint

   Check the pods ip.
10. ConfigMap and Secret

    ConfigMap and Secret are using key-value, the difference is that Secret use base64 secret the value, ConfigMap's value can be any value.

Can write ConfigMap and Secret in the yaml and pod based on the template.

11. DaemonSet

    Run pod in every node. When you add or remove a node, the pod will create or delete it in the node automation.

    You can use `kubectl rollout history` see the DaemonSet version
12. Job and CronJob

    The job will create a pod when the condition satisfy and stop based on the completions (pod number), activeDeadlineSeconds (run time), and ttlSecondsAfterFinished (After finish time)

    Cron Job will create a job and pod based on the Schedule. Cron Schedule Syntax
13. ETCD Backing up and Restoring

    Backing up:

    ```
    ETCDCTL_API=3 etcdctl --cacert=/opt/kubernetes/ssl/ca.pem --cert=/opt/kubernetes/ssl/server.pem --key=
    /opt/kubernetes/ssl/server-key.pem --endpoints 192.168.1.36:2379 snapshot save snapshotdb

    etcdctl version + etcdctl + certificate + --endpoint + snapshot save + file position.
    ```

    Restoring:

    ```
    ETCDCTL_API=3 etcdctl --endpoints 192.168.1.36:2379 snapshot restore snapshotdb
    ```

K8s RBAC

1. User Account vs Service Account

   User accounts are for humans. Service accounts are for processes, which run in pods.

   User accounts are intended to be global...Service accounts are namespaced.
2. ClusterRole vs Role

   ClusterRole isn't in the namespace, Role is in the namespace.

   ClusterRoleBinding can't bind the Role to the user group and service account.
   RoleBinding can bind the ClusterRole.
3. Create a certificate

   a. Generate key.

   ```
   openssl genrsa -out k8s.key 2048
   ```

   b. Generate csr and assign user/group, cn is user, o is group.

   ```
   openssl genrsa -out k8s.key 2048
   ```

   c. Use ca and cakey to generate the crt.

   ```
   openssl x509 -req -in k8s.csr -CA /etc/kubernetes/pki/ca.crt -CAkey /etc/kubernetes/pki/ca.key -
   CAcreateserial -out k8s.crt -days 365
   ```

   d. update k8s config file.

   ```
   openssl x509 -req -in k8s.csr -CA /etc/kubernetes/pki/ca.crt -CAkey /etc/kubernetes/pki/ca.key -
   CAcreateserial -out k8s.crt -days 365
   ```

4. Create Role and Cluster Role
5. Create RoleBinding and ClusterRoleBinding

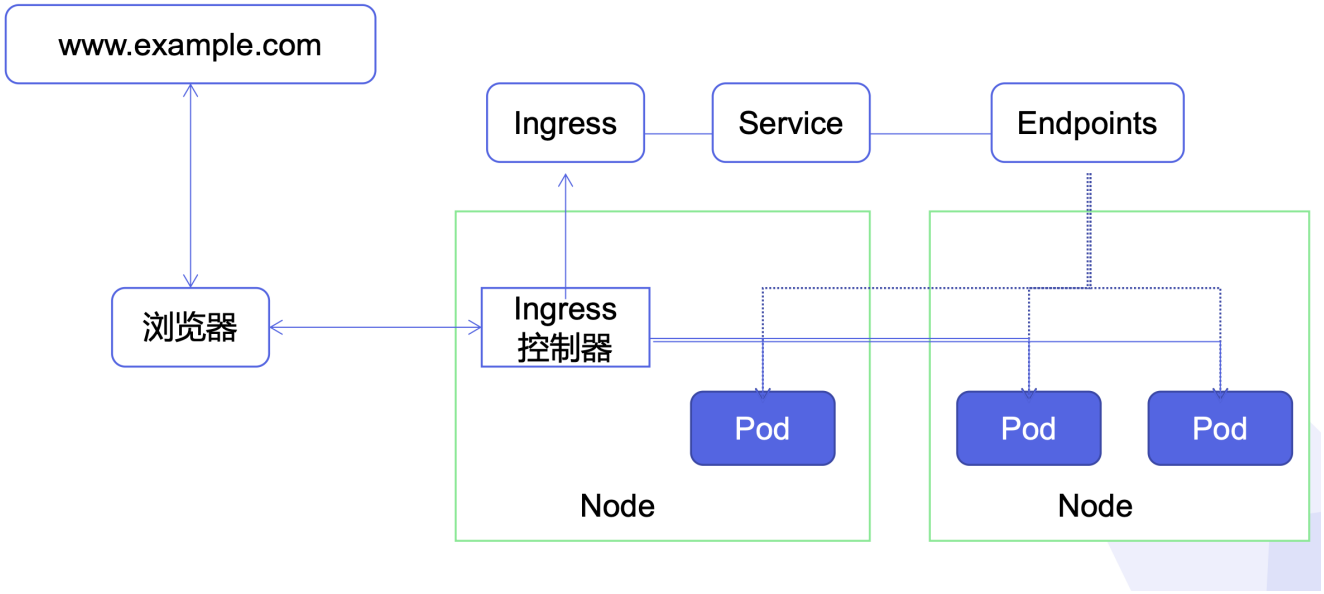
kubectl commands

> *K8s command format:*
> *kubectl + operate + resource type + resource name + other params.*
> *If no resource name, the kubectl will show all.*

k8s visit check

> *If link don't work, please use this flow to check the visit*

www.example.com--(DNS)--IP address-->firewall--->balance--->Ingress Controller--(Ingress)---Service--(name)--Endpoint--(label)--Pod---(volume)--PVC-(volumeNmae)---PV--- volume pluginNFS----volumefolder)



tips:

> *add the ip and name in the /etc/host, you can use `ssh name` visit, the user is the same as your current.*

> *`ipvsadm -Ln`: This can see the IP mapping.*

> *vs code can ssh the remote and do something.*