k8s-docker-2-containerd

author: Uncle Dragon
date: 2021-06-21

如何迁移

迁移

- 驱逐节点
- 停止 `kubelet` 和 `docker` 服务
- 配置 `containerd`
- 重启 `containerd`
- 修改 `runtime`
- 启动 `kubelet`
- 检查
- 恢复节点
- 删除 `docker` (非必需)

如何迁移

1. 我们要检查当前的运行的容器时什么

```
$ kubectl get nodes -o wide
NAME          STATUS   ROLES    AGE    VERSION   INTERNAL-IP
EXTERNAL-IP   OS-IMAGE             KERNEL-VERSION
CONTAINER-RUNTIME
test-master   Ready    master   307d   v1.18.8   10.8.30.157
<none>        Ubuntu 18.04.5 LTS   4.15.0-122-generic
docker://19.3.12
test-n1       Ready    <none>   277d   v1.18.8   10.8.30.152
<none>        Ubuntu 18.04.4 LTS   4.15.0-128-generic
docker://19.3.12
test-n2       Ready    <none>   307d   v1.18.8   10.8.30.156
<none>        Ubuntu 18.04.4 LTS   4.15.0-128-generic
docker://19.3.12
test-n3       Ready    <none>   307d   v1.18.8   10.8.30.155
<none>        Ubuntu 18.04.5 LTS   4.15.0-142-generic
docker://19.3.12
test-n4       Ready    <none>   307d   v1.18.8   10.8.30.161
<none>        Ubuntu 16.04.6 LTS   4.15.0-122-generic
docker://19.3.12
test-n5       Ready    <none>   307d   v1.18.8   10.8.30.158
<none>        Ubuntu 20.04.2 LTS   5.4.0-73-generic
docker://20.10.6
```

2. 检查 containerd cli ctr 是否存在

```
ctr version
Client:
  Version:  1.4.6
  Revision: d71fcd7d8303cbf684402823e425e9dd2e99285d
  Go version: go1.13.15

Server:
  Version:  1.4.6
  Revision: d71fcd7d8303cbf684402823e425e9dd2e99285d
  UUID: 9ee94062-6162-4044-aba7-8b2dc2431cd9
```

3. 检查 moby 命名空间

```
# ctr namespace list
NAME LABELS
```

```
moby


# ctr -n moby c ls
CONTAINER
      IMAGE    RUNTIME
02260fbe3748f4043d39a47105aca8a3aa6bef5b5dc35c2e01e5a95d2df6bf44    -        io.containerd.runtime.v1.linux
0827c2390d3f824d189662f3f30199c6d712b46f32cde3718457bd8d0fee473d    -        io.containerd.runtime.v1.linux
097c3b72a7d74a4484097487dfe92ca8136202bf581d7087b8aca234d3095ec3    -        io.containerd.runtime.v1.linux
20d90cd5d2cbba75c110f3071498ccfb2546d4e2a26485ace7123a3801150624    -        io.containerd.runtime.v1.linux
34360aa18c08dd552b65e21a32c7189c85ceed8d95a8ab19dad26aa4e25ebf8b    -        io.containerd.runtime.v1.linux
37413d2581f895f99f7acc14e5e5a9b87dfb804f9176726f6639b2a43abdf72c    -        io.containerd.runtime.v1.linux
396b1877e2d291e845d036a0a8d5a0a72995c716e72dd0766f742f523d4decfe    -        io.containerd.runtime.v1.linux
3ddb8a34a401acceef83a3749963d6c43508e1b2be33e056efe2f4242d8a47a5    -        io.containerd.runtime.v1.linux
3ea03d2796b62e3286b9cf2b30f0902ec35df89af0884a8e0670b22f727fc96b    -        io.containerd.runtime.v1.linux
409e8c0b76bb0a51600c6081ee7d4d1502af5f6a38f60dc077eb7ce384d4ba49    -        io.containerd.runtime.v1.linux
5cee70ca1ad365d3dc326c97e3b53b87353ee3835af98244aea0f72e1a10a0e0    -        io.containerd.runtime.v1.linux
66c1371599b4b2cd4fb6ce27147bbab4348054745121c4fe9a64bdffa5a3e57b    -        io.containerd.runtime.v1.linux
79d87cf7e89d8029f85a0ee207837bd102c65a6c4dca840b81ac895dff97d4ed    -        io.containerd.runtime.v1.linux
8599a6c45eb7d66c0d74d69000c721f33d95a2c1b2e6e813d8ef1dcd1ca4e1c7    -        io.containerd.runtime.v1.linux
89e947b1f203c2d55894a33c3c01f0b734b08a714cc2446e45c7df12bcdcc227    -        io.containerd.runtime.v1.linux
902112cfb762b60d353b9ab6e104bcb4ef6f626db10b56d8347450044183460 7    -        io.containerd.runtime.v1.linux
9c30b5e7d0f75a341194f612380e3e2352b6ad6ee63623aafc3d19e3f2bb1a59    -        io.containerd.runtime.v1.linux
9f76f8664b25a545253aad1a6b11731dc5569215dbf8b0a614d91bd1f181d552    -        io.containerd.runtime.v1.linux
a2e5cdfc95a34bd1db0049476282baefa955c58600b2bc323ff7b1d4bc060be7    -        io.containerd.runtime.v1.linux
a39582855bcc4fd04523ce03de5e7bd7202f0c2b66534606bc0c0419cf661483    -        io.containerd.runtime.v1.linux
a7c1dfea50841b3a8f6644f5c5902f61db71f80ec2c80ac64720b5a757cdc8b5    -        io.containerd.runtime.v1.linux
```

```
bde73ac0cc419904a8022bae23db0376e0521ba4888fd384bd3b063f88941482    -        io.containerd.runtime.v1.linux
d1a86b65635357a3e3346d54df85c7c9c040105af97b97176ca4b8d0400498cf    -        io.containerd.runtime.v1.linux
d239c43b914e7c464bf55c25cbc2e55596d35c89ecf70ac60a344c57b151d081    -        io.containerd.runtime.v1.linux
d46bd5ce29c5d4b1137d1840e7a8412e6ed7d058a421a4c8df9ebfd5eb707ad5    -        io.containerd.runtime.v1.linux
d4e8bef66fe4be1483fd32bf7bdce93166c69d21f430d3a746d3766410f93b48    -        io.containerd.runtime.v1.linux
d5c9fb0a307ef49e90a680cf6b3865cf2323960f02b5d42a6e6f54964c3ab830    -        io.containerd.runtime.v1.linux
db8efd6e685d095a2578b9399e3495d08f7a47750b406113439f3175fb3bc0a5    -        io.containerd.runtime.v1.linux
e5db513a09db4fe1d82f9b00564e79450b898254a846782406d68736e7b84897    -        io.containerd.runtime.v1.linux
e671b2d14830d9d6a64df18599929ba4e3926f394148929389f7c482cfea45db    -        io.containerd.runtime.v1.linux
f0fbb7a25a040c309eb4bfcd08e08e5265349537d3a47387f02f5f051a3d3d0f    -        io.containerd.runtime.v1.linux
f86c123dec237cf02706aaf8cad0b7753fa2323f7588aacf2aa64037363c60ee    -        io.containerd.runtime.v1.linux

```

一切正常，那么我们可以更换cri 了，每次一个节点，首先是工作节点，然后是控制节点，如果只有一个控制节点，那么你将暂时会失去对集群的访问，正常情况下会自动恢复的。

迁移

我们 从 `test-n4` 开始：

驱逐节点

```
# 设置节点不可调度
$ kubectl cordon test-n4
node/test-n4 cordoned

# 将节点上资源调度到其他节点
$ kubectl drain test-n4 --delete-local-data --force --ignore-daemonsets
node/test-n4 already cordoned
```

```
WARNING: ignoring DaemonSet-managed Pods: kube-eye/fluentd-es-v3.1.0-4j5mr, kube-eye/kube-eye-worker-q76xt, kube-system/kube-flannel-ds-amd64-xflkq, kube-system/kube-proxy-dpwl4, kubeedge/iptables-zgldl, kubesphere-logging-system/fluent-bit-lgc5v, kubesphere-monitoring-system/node-exporter-7xd84, weave/weave-scope-agent-hr4zx
evicting pod anxinyun/anxinyun-config-center-65b7f88d5c-62zbk
evicting pod anxinyun/testxf-console-574775bb5b-dnnqv
evicting pod kubesphere-logging-system/elasticsearch-logging-discovery-1
evicting pod savoir/savoir-bigscreen-67989dccc8-l26n5
...
...
I0621 15:36:02.908180   13546 request.go:621] Throttling request took 1.199580189s, request: POST:https://10.8.30.157:6443/api/v1/namespaces/savoir/pods/savoir-webapi-6f677566b5-nt7ls/eviction
pod/anxinyun-config-center-65b7f88d5c-62zbk evicted
I0621 15:36:13.006128   13546 request.go:621] Throttling request took 5.254427794s, request: GET:https://10.8.30.157:6443/api/v1/namespaces/anxinyun/pods/testxf-webapi-674b57b8d6-2896v
pod/logsidecar-injector-deploy-74c66bfd85-xsvcv evicted
pod/iota-mqtt-5dd58b8b84-rtlvb evicted
pod/iota-web-576b4965fc-99cm2 evicted
pod/flink-taskmanager-665b77b674-k5xsg evicted
pod/anxinyun-master-7974dfd46d-ttxxs evicted
pod/elasticsearch-logging-discovery-1 evicted
...
...
node/test-n4 evicted
```

```
$ kubectl get node
NAME          STATUS                     ROLES    AGE    VERSION
test-master   Ready                      master   307d   v1.18.8
test-n1       Ready                      <none>   277d   v1.18.8
test-n2       Ready                      <none>   307d   v1.18.8
test-n3       Ready                      <none>   307d   v1.18.8
test-n4       Ready,SchedulingDisabled   <none>   307d   v1.18.8
test-n5       Ready                      <none>   307d   v1.18.8
```

下面连接到 `test-n4`

停止 `kubelet` 和 `docker` 服务

```
$ systemctl stop kubelet
$ systemctl stop docker
```

配置 containerd

注释掉 /etc/containerd/config.toml 中的 disabled_plugins 行

或者 重新生成一个新的配置文件

```
$ containerd config default > /etc/containerd/config.toml
```

重启 containerd

```
systemctl restart containerd
```

修改 runtime

编辑 /var/lib/kubelet/kubeadm-flags.env ，增加 --container-runtime=remote 和 --container-runtimeendpoint=unix:///run/containerd/containerd.sock

```
$ vi /var/lib/kubelet/kubeadm-flags.env

KUBELET_KUBEADM_ARGS="--cgroup-driver=cgroupfs --network-plugin=cni --pod-infra-container-image=registry.aliyuncs.com/google_containers/pause:3.2"
```

```
KUBELET_KUBEADM_ARGS="--cgroup-driver=cgroupfs --network-plugin=cni --pod-infra-container-image=registry.aliyuncs.com/google_containers/pause:3.2 --container-runtime=remote --container-runtime-endpoint=unix:///run/containerd/containerd.sock"
```

启动 kubelet

```
systemctl start kubelet
```

检查

```
$ kubectl get node -o wide
NAME          STATUS                     ROLES    AGE    VERSION
INTERNAL-IP   EXTERNAL-IP   OS-IMAGE             KERNEL-VERSION
  CONTAINER-RUNTIME
test-master   Ready                      master   307d   v1.18.8
10.8.30.157   <none>        Ubuntu 18.04.5 LTS   4.15.0-122-generic
  docker://19.3.12
test-n1       Ready                      <none>   277d   v1.18.8
10.8.30.152   <none>        Ubuntu 18.04.4 LTS   4.15.0-128-generic
  docker://19.3.12
test-n2       Ready                      <none>   307d   v1.18.8
10.8.30.156   <none>        Ubuntu 18.04.4 LTS   4.15.0-128-generic
  docker://19.3.12
test-n3       Ready                      <none>   307d   v1.18.8
10.8.30.155   <none>        Ubuntu 18.04.5 LTS   4.15.0-142-generic
  docker://19.3.12
test-n4       Ready,SchedulingDisabled   <none>   307d   v1.18.8
10.8.30.161   <none>        Ubuntu 16.04.6 LTS   4.15.0-122-generic
  containerd://1.2.13
test-n5       Ready                      <none>   307d   v1.18.8
10.8.30.158   <none>        Ubuntu 20.04.2 LTS   5.4.0-73-generic
  docker://20.10.6
```

可以看到 `containerd` 是我们刚刚更改的节点的运行时了

恢复节点

现在可以把节点恢复调度了

```
$ kubectl uncordon test-n4
node/test-n4 uncordoned

$ kubectl get node -o wide
NAME          STATUS   ROLES    AGE    VERSION   INTERNAL-IP
EXTERNAL-IP   OS-IMAGE             KERNEL-VERSION       CONTAINER-
RUNTIME
test-master   Ready    master   307d   v1.18.8   10.8.30.157
<none>        Ubuntu 18.04.5 LTS   4.15.0-122-generic
docker://19.3.12
test-n1       Ready    <none>   277d   v1.18.8   10.8.30.152
<none>        Ubuntu 18.04.4 LTS   4.15.0-128-generic
docker://19.3.12
test-n2       Ready    <none>   307d   v1.18.8   10.8.30.156
<none>        Ubuntu 18.04.4 LTS   4.15.0-128-generic
docker://19.3.12
test-n3       Ready    <none>   307d   v1.18.8   10.8.30.155
<none>        Ubuntu 18.04.5 LTS   4.15.0-142-generic
docker://19.3.12
test-n4       Ready    <none>   307d   v1.18.8   10.8.30.161
<none>        Ubuntu 16.04.6 LTS   4.15.0-122-generic
containerd://1.2.13
test-n5       Ready    <none>   307d   v1.18.8   10.8.30.158
<none>        Ubuntu 20.04.2 LTS   5.4.0-73-generic
docker://20.10.6

```

删除 `docker` (非必需)

如果不用 `docker` 可以把它卸载了

```
$ apt-get purge docker-ce docker-ce-cli
```

现在我们已经成功的把工作节点 `test-n4` 的容器运行时切换到了 `containerd`，下面继续下一个节点重复上述步骤。