2.1.1 OpenShift 3 的私有云部署

1. 部署环境准备

在私有云部署，首先要完成部署环境的准备。包含内容如下：

（1） 虚拟机创建

根据高可用部署架构置备对应数目的虚拟机，3 个 Master 节点、3 个 Infra 节点、2 个 app 节点、2 个负载均衡器节点，由于演示环境资源有限，部分非关键角色将共用虚拟机，在实际企业部署时不推荐这么做。

虚拟机资源配置及功能说明如下表 2-9 所示：

表 2-9 OpenShift 集群资源配置说明表

名称	域名	IP 地址	CPU/内存/操作系统盘	额外裸磁盘	节点角色
Master01	master01.example.com	192.168.0.4	2C/8G/100G		OpenShift-Master
					Ansible
					ETCD
Master02	master02.example.com	192.168.0.5	2C/8G/100G		OpenShift-Master
					外部镜像库
					Yum
					ETCD
Master03	master03.example.com	192.168.0.6	2C/8G/100G		OpenShift-Master
					外部镜像库
					ETCD
Infra01	infra01.example.com	192.168.0.7	4C/16G/100G	100G	OpenShift-Node
					内部镜像库
					Router
					EFK+Metrics

Infra02	infra02.example.com	192.168.0.8	4C/16G/100G	100G	OpenShift-Node 内部镜像库 Router EFK+Metrics
Infra03	infra03.example.com	192.168.0.9	4C/16G/100G	100G	OpenShift-Node 内部镜像库 Router EFK+Metrics
Node01	node01.example.com	192.168.0.10	8C/32G/100G	100G	OpenShift-Node
Node02	node02.example.com	192.168.0.11	8C/32G/100G	100G	OpenShift-Node
LB01	lb01.example.com	192.168.0.12	2C/4G/60G		LB + keepalived
LB02	lb02.example.com	192.168.0.13	2C/4G/60G		LB + keepalived
VIP1	master.example.com	192.168.0.1			Master VIP
VIP2	*.apps.example.com	192.168.0.2			Router VIP
VIP3	registry.example.com	192.168.0.3			Registry VIP

（2） 操作系统配置

所有节点均使用 **Red Hat Enterprise Linux 7.6**，并选择 **Minimal** 模式进行安装，并完成必要的操作系统配置，如网络、主机名、配置时区、时间同步、关闭防火墙 Firewalld、开启 Selinux、启动 NetworkManager 服务，具体操作命令行这里不再列出。

注意需要提供节点之间的域名解析，可以通过 DNS 实现或者本地 hosts 文件实现，除了节点之外还需要包含 Master VIP 和 Registry VIP 的解析。

（3） Yum 源配置

在 Master02 节点配置 Yum 服务，并将所有虚拟机的 Yum 源指向 Master02 节点，具体操作过程略。安装社区版和企业版的 RPM 包不同，需要的 RPM 频道查看官方文档获取。

需要注意的是，虽然我们已经在操作系统配置时关闭了 Firewalld 服务，但是在安装 OpenShift 过程中会启动防火墙 iptables，由于与 OpenShift 共用节点，导致在安装过程中无法访问 Yum 仓库安装软件包而安装失败，可以在这里提前将允许访问 Yum 端口的 iptables 规则添加。

（4） 外部镜像仓库

在 Master02 和 Master03 安装 docker-distribution，配置镜像仓库：

yum -y install docker-distribution

systemctl start docker-distribution

systemctl enable docker-distribution

在通过 LB01 和 LB02 实现外部镜像仓库的高可用，绑定 Registry VIP：192.168.0.3，操作过程见后文。

最后需要根据官方给出的安装所需要的镜像列表，在连接公网的条件下拉取镜像后导入外部镜像仓库中。

如果镜像仓库使用非安全协议 HTTP 的话，需要在所有节点安装 docker，在 docker 配置文件/etc/sysconfig/docker 中添加非安全的镜像仓库参数--inseccure-registry registry.example.com，然后重新启动 docker 进程。

与 Yum 仓库同样的问题，由于与 OpenShift 共用节点，导致在安装过程中 iptables 会被启动，导致无法访问镜像仓库拉取镜像而安装失败，可以在这里提前将允许访问镜像仓库端口的 iptables 规则添加。

2. OpenShift 3 安装前准备

（1） 安装必要基础软件包

安装软件包并配置基础环境。在所有 OpenShift 节点上安装 OpenShift 需要的软件包，命令如下：

yum -y install wget git net-tools bind-utils iptables-services bridge-utils bash-completion vim atomic-openshift-excluder atomic-openshift-docker-excluder unzip atomic-openshift-utils

yum -y update

atomic-openshift-excluder unexclude

（2） 安装和配置 Docker

在所有节点 OpenShift 节点上安装 Docker。命令如下：

yum install -y docker

我们为 app 节点额外配置的 100G 的裸磁盘作为 docker stoage。以一个节点为例说明配置过程，其余节点操作相同。

修改 docker storage 配置文件

vi /etc/sysconfig/docker-storage-setup

STORAGE_DRIVER=overlay2

DEVS=/dev/vdb

CONTAINER_ROOT_LV_NAME=docker-lv

CONTAINER_ROOT_LV_SIZE=100%FREE

CONTAINER_ROOT_LV_MOUNT_PATH=/var/lib/docker

VG=docker-vg

执行 docker-storage-setup 命令，配置 docker storage

docker-storage-setup

上述命令会将/dev/sdb 建立逻辑卷/dev/mapper/docker--vg-docker--lv，并挂在到/var/lib/docker 目录下，如下图 2-44 所示：

lsblk

```
[root@ubmpapld00008 ~]# lsblk
NAME                      MAJ:MIN RM SIZE RO TYPE MOUNTPOINT
vda                       252:0    0  40G  0 disk
└─vda1                    252:1    0  40G  0 part /
vdb                       252:16   0  50G  0 disk
└─vdb1                    252:17   0  50G  0 part
  └─docker--vg-docker--lv 253:0    0  50G  0 lvm  /var/lib/docker
```

图 2-44 docker storage

如果还需要修改 docker 配置文件，如添加非安全的镜像仓库、修改 docker0 桥的地址等，则在启动 docker 服务前进行。

启动 docker 进程并开机自启

systemctl start docker

systemctl enable docker

（3） 安装和配置 Ansible 主控节点

OpenShift 3 使用 Ansible 完成自动化安装，关于 Ansible 的介绍和使用在本书的第五章节，不熟悉 Ansible 的读者可先阅读第五章 Ansible 小节。

在 master01 节点安装 ansible 和用于安装 OpenShift 的 playbook

yum -y install openshift-ansible

使用 Ansible 需要通过 SSH 连接到被控节点，可以配置 SSH 免密登录。

在 master01 节点上生成 SSH 公私钥。命令如下，应答输入请直接输入回车。

ssh-keygen

在 master01 节点上配置 master01 节点到所有节点的 SSH 主机互信。命令如下，请根据提示输入远程主机 root 账户密码。

ssh-copy-id root@<hostname 或 ip>

（4） 临时配置 Master VIP

在高可用架构下，访问 Master 节点需要经过负载均衡，但是往往在安装之前负载均衡未必可以配置，这时就需要我们临时将 Master 的 VIP 绑定在某个节点上，等集群安装完成后再有 Keepalived 接管 VIP。这里我们选择将 VIP 绑定在 lb01 节点上，操作过程如下：

cd /etc/sysconfig/network-scripts

cp ifcfg-ens192 ifcfg-ens192:0

vi ifcfg-ens192:0

删除 UUID，修改 IP 地址为 MasterVIP，启动网卡 ifcfg-ens192:0

ifup ens192:0

确认分配的 VIP 已经绑定在网卡上

ip a

3. OpenShift 3 安装

在 master01 节点修改 ansible inventory 文件/etc/ansible/hosts，内容如下：

```
[OSEv3:children]
masters
nodes
etcd
lb


[OSEv3:vars]
#Basic
ansible_ssh_user=root
openshift_deployment_type=openshift-enterprise
openshift_release=v3.11
openshift_image_tag=v3.11.98
openshift_pkg_version=-3.11.98


oreg_url=registry.example.com/openshift3/ose-${component}:${version}
openshift_examples_modify_imagestreams=true
openshift_master_identity_providers=[{'name': 'htpasswd_auth',
'login': 'true', 'challenge': 'true', 'kind':
'HTPasswdPasswordIdentityProvider'}]
openshift_docker_options="--selinux-enabled --insecure-registry
172.30.0.0/16 --log-driver json-file --log-opt max-size=50M --log-opt
max-file=3 --insecure-registry registry.example.com  --add-registry
registry.example.com"


openshift_disable_check=docker_image_availability,docker_storage,mem
ory_availability,disk_availability


openshift_master_cluster_method=native
openshift_master_cluster_hostname=master.example.com
```

```
openshift_master_cluster_public_hostname=master.example.com

openshift_master_default_subdomain=apps.example.com

os_sdn_network_plugin_name='redhat/openshift-ovs-multitenant'


#Certification expire

openshift_hosted_registry_cert_expire_days=3650

openshift_ca_cert_expire_days=3650

openshift_node_cert_expire_days=3650

openshift_master_cert_expire_days=3650

etcd_ca_default_days=3650




#router

openshift_hosted_router_replicas=3

openshift_router_selector='infra=true'


#registry

openshift_registry_selector='infra=true'

openshift_hosted_registry_replicas=3


#service catalog

openshift_enable_service_catalog=false

ansible_service_broker_install=false


#monitoring

openshift_cluster_monitoring_operator_install=true

openshift_metrics_install_metrics=true

openshift_metrics_cassandra_replicas=3


#logging
```

```
   openshift_logging_install_logging=true
   openshift_logging_es_nodeselector={"infra":"true"}
   openshift_logging_es_cluster_size=3


   openshift_node_groups=[{'name': 'node-config-master', 'labels':
['node-role.kubernetes.io/master=true']}, {'name': 'node-config-infra',
'labels': ['node-role.kubernetes.io/infra=true', 'infra=true']},
{'name': 'node-config-compute', 'labels': ['node-
role.kubernetes.io/compute=true']}]

   # host group for masters
   [masters]
   master01.example.com
   master02.example.com
   master03.example.com

   # host group for etcd
   [etcd]
   master01.example.com
   master02.example.com
   master03.example.com

   # master loadbalancer
   [lb]
   lb01.example.com
   lb02.example.com

   # host group for nodes, includes region info
   [nodes]
```

```
master01.example.com  openshift_node_group_name='node-config-master'
master02.example.com  openshift_node_group_name='node-config-master'
master03.example.com  openshift_node_group_name='node-config-master'
infra01.example.com  openshift_node_group_name='node-config-infra'
infra02.example.com  openshift_node_group_name='node-config-infra'
infra03.example.com  openshift_node_group_name='node-config-infra'
node01.example.com  openshift_node_group_name='node-config-compute'
node02.example.com  openshift_node_group_name='node-config-compute'
```

由于篇幅有限，我们就不解释每个参数表示的含义了，感兴趣的读者参考官网链接 https://docs.openshift.com/container-platform/3.11/install/configuring_inventory_file.html。

安装前测试节点连通性

ansible -m ping all

执行安装

cd /usr/share/ansible/openshift-ansible

ansible-playbook -vv playbooks/prerequisites.yml

ansible-playbook -vv playbooks/deploy_cluster.yml

如安装中遇到问题需卸载可以执行：

ansible-playbook -vv playbooks/adhoc/uninstall.yml

安装完成后通过执行如下命令执行检测集群状态。

- 执行 oc get node 可以查看集群节点的情况。
- 执行 oc get pod --all-namespaces 检测集群所有容器是否运行正常。

4. 安装并配置软负载和 Keepalived

在 OpenShift 安装过程中，已经把软负载均衡软件 Haproxy 安装在 lb01 和 lb02 节点上了，但是仅配置了 Master 的负载均衡。这里由于我们共用负载均衡节点，所以需要完成其他两个 VIP 以及 Keepalived 的配置。

两个 LB 节点执行如下命令安装软件包：

yum -y install haproxy keepalived psmisc

（1） 配置 Haproxy

修改 haproxy 配置文件/etc/haproxy/haproxy.conf，内容如下：

```
# Global settings
#---------------------------------------------------------------------
global
    maxconn     20000
    log         /dev/log local0 info
    chroot      /var/lib/haproxy
    pidfile     /var/run/haproxy.pid
    user        haproxy
    group       haproxy
    daemon

    # turn on stats unix socket
    stats socket /var/lib/haproxy/stats

#---------------------------------------------------------------------
# common defaults that all the 'listen' and 'backend' sections will
# use if not designated in their block
#---------------------------------------------------------------------
defaults
    mode                    http
    log                     global
    option                  httplog
    option                  dontlognull
    option forwardfor       except 127.0.0.0/8
    option                  redispatch
    retries                 3
    timeout http-request    10s
    timeout queue           1m
    timeout connect         10s
    timeout client          300s
    timeout server          300s
    timeout http-keep-alive 10s
    timeout check           10s
    maxconn                 20000
listen stats :1936
    mode http
    monitor-uri /healthz
    stats enable
    stats hide-version
    stats realm Haproxy\Statistics
    stats uri /
    stats auth admin:admin

frontend  docker-distribution
```

```
    bind *:5000
    default_backend docker-distribution-instance
    mode http
    option httplog

frontend    atomic-openshift-api
    bind *:8443
    default_backend atomic-openshift-api
    mode tcp
    option tcplog

frontend    router-http-apps
    bind *:80
    default_backend router-http-server
    mode tcp
    option tcplog

frontend    router-https-apps
    bind *:443
    default_backend router-https-server
    mode tcp
    option tcplog

backend docker-distribution-instance
    balance source
    mode http
    server          registry0 192.168.0.5:5000 check
    server          registry1 192.168.0.6:5000 check


backend atomic-openshift-api
    balance source
    mode tcp
    server          master0 192.168.0.4:8443 check
    server          master1 192.168.0.5:8443 check
    server          master2 192.168.0.6:8443 check

backend router-http-server
    balance source
    mode tcp
    server    router1 192.168.0.7:80 check
    server    router2 192.168.0.8:80 check
    server    router3 192.168.0.9:80 check

backend router-https-server
    balance source
    mode tcp
    server    router1 192.168.0.7:443 check
    server    router2 192.168.0.8:443 check
```

```
    server    router2192.168.0.9:443 check
```

（2） 配置 Keepalived

Keepalived 主要实现 VIP 的管理，该软件使用多播实现心跳，节点网络（如交换机）需要支持组播通信，该软件实现的原理和细节请读者自行查阅网络资料学习，这里不多赘述。修改 lb01 节点的 Keepalived 配置文件/etc/keepalived/keepalived.conf，内容如下：

```
! Configuration File for keepalived
vrrp_script chk_haproxy {
    script "killall -0 haproxy"
    interval 2
    weight -2
    fail 2
    rise 1
}

# master VIP
vrrp_instance VI_1 {
    state MASTER     #对于 master VIP，lb01 为主，lb02 为备（BACKUP）
    interface ens192     #绑定 VIP 的网卡名称
    virtual_router_id 50     #主备必须一致
    priority 101              #lb01 为主，优先级为 101；lb02 为备，优先级为 100
    advert_int 1
    authentication {
        auth_type PASS
        auth_pass 1111     #主备密码必须一致
    }
    virtual_ipaddress {
        192.168.0.1        #master VIP
    }
    track_script {
       chk_haproxy
    }
}

# Registry VIP
vrrp_instance VI_2 {
    state MASTER     #对于 Registry VIP，lb01 为主，lb02 为备（BACKUP）
    interface ens192     #绑定 VIP 的网卡名称
    virtual_router_id 51     #主备必须一致
    priority 101              #lb01 为主，优先级为 101；lb02 为备，优先级为 100
    advert_int 1
    authentication {
```

```
        auth_type PASS
        auth_pass 2222    #主备密码必须一致
    }
    virtual_ipaddress {
        192.168.0.3       #Registry VIP
    }
    track_script {
       chk_haproxy
    }
}

# Router VIP
vrrp_instance VI_3 {
    state BACKUP    #对于 Router VIP，lb01 为备，lb02 为主（MASTER）
    interface ens192    #绑定 VIP 的网卡名称
    virtual_router_id 52    #主备必须一致
    priority 100            #lb01 为备，优先级为 100；lb02 为主，优先级为 101
    advert_int 1
    authentication {
        auth_type PASS
        auth_pass 3333    #主备密码必须一致
    }
    virtual_ipaddress {
        192.168.0.2       #Registry VIP
    }
    track_script {
       chk_haproxy
    }
}
```

可以看到，master VIP 和 registry VIP 的主是 lb01，也就是默认情况下，这两个 VIP 会绑定在 lb01 节点上，除非 lb01 节点 haproxy 进程故障，这两个 VIP 才会漂移到 lb02 上；routerVIP 的主是 lb02，默认在 lb02 上。

lb02 节点的 Keepalived 配置文件这里就不提供了，根据提供的 lb01 节点的配置和说明可以很容易的写出 lb02 的 Keepalived 配置文件。

（3） 配置防火墙

如果 lb01 和 lb02 节点的防火墙是关闭的，以下步骤可以忽略，建议将节点的防火墙打开提高安全性，需要在两个 lb 节点完成以下操作。

安装软件启动服务

yum -y install iptables iptables-services

systemctl start iptables

systemctl enable iptables

配置规则

iptables -I INPUT -p tcp -m state --state NEW -m tcp --dport 8443 -j ACCEPT

iptables -I INPUT -p tcp -m state --state NEW -m tcp --dport 5000 -j ACCEPT

iptables -I INPUT -p tcp -m state --state NEW -m tcp --dport 80 -j ACCEPT

iptables -I INPUT -p tcp -m state --state NEW -m tcp --dport 443 -j ACCEPT

iptables -I INPUT -i ens192 -d 224.0.0.18/32 -j ACCEPT

service iptables save

（4） 启动服务

先启动 haproxy 服务，并开机自启

systemctl start haproxy

systemctl enable haproxy

在启动 Keepalived 服务之前，需要先将临时绑定的 master VIP 解除，否则会出现 IP 冲突。在 lb01 节点执行如下操作。

ifdown ens192:0

rm -rf /etc/sysconfig/network-scripts/ifcfg-ens192:0

启动 Keepalived 服务并开机自启

systemctl start keepalived

systemctl enable keepalived

配置完成后，执行 ip a 检测 VIP 是否绑定成功，并且通过分别停止 lb01 和 lb02 节点的 haproxy 服务测试 VIP 是否会符合预期的飘动。

5. OpenShift 3 安装后的配置

（1） 配置外部 DNS 解析

在安装完成之后，需要将需要外部访问的域名添加到 DNS server 中，这样客户端才能访问管理控制台和运行在 OpenShift 上的应用，需要解析的 DNS A 记录有如下两条：

192.168.0.1 master.example.com

192.168.0.2 *.apps.example.com

（2） 添加用户

在集群安装完成后，需要添加用户才能登陆使用 OpenShift。OpenShift 支持集成多种用户认证系统，如 LDAP、Github 等，我们将在第三章详细说明这部分。在本示例中，我们使用 HTPasswdPasswordIdentityProvider 的认证方式。

添加管理员用户 admin，并设置密码为 admin，赋予集群管理员权限。在 master01 上执行如下命令：

ansible -m shell -a 'htpasswd -bc /etc/origin/master/htpasswd admin admin' masters

oc adm policy add-cluster-role-to-user cluster-admin admin

如果需要添加其他用户，重复执行第一条 Ansible 语句即可。

（3） 内部镜像仓库配置持久化存储

后端存储我们选择 NAS 提供，需要为内部镜像仓库创建一个卷，假设为/data/internal-registry。持久化内部 docker registry 需要创建 PV 和 PVC，文件内容如下：

cat intermal-registry-pvc.yaml

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: registry-claim
  namespace: default
spec:
  accessModes:
  - ReadWriteMany
  resources:
```

```
    requests:
      storage: 200Gi
```

cat internal-registry-pv.yaml

```
apiVersion: v1
kind: PersistentVolume
metadata:
  name: registry-volume
spec:
  accessModes:
  - ReadWriteMany
  capacity:
    storage: 200Gi
  nfs:
    path: /data/internal-registry
    server: 10.1.0.1
  persistentVolumeReclaimPolicy: Retain
```

在 master01 节点执行操作，切换到 default 项目下，挂载卷到 docker-registry Pod 中。

oc project default

oc create -f internal-registry-pv.yaml

oc create -f intermal-registry-pvc.yaml

oc set volume dc/docker-registry --add --overwrite --name=registry-storage --type=persistentVolumeClaim --claim-name=registry-claim

（4） 监控配置持久化存储

监控的持久化配置包含 Hawkular 监控系统下的 Cassandra 和 Prometheus 监控系统下的 Alertmanager、Prometheus。

Cassandra 持久化的配置方法与 docker-registry 完全一致，分别创建 PV 和 PVC，然后挂载到 Cassandra Pod 中，如果 Cassandra 使用集群部署，则需要为每个实例分别创建独立的卷。

Prometheus 的挂载与 Cassandra 略有区别，主要在于 Prometheus 使用 Operator 完成部署，PVC 的创建时由 Operator 完成的，我们只需要按需要的模式和大小提前创建出 PV 就可以了，Operator 已经为我们挂载好了。默认需要的 PV 列表如下：

用于 Prometheus 的两个 PV：大小 50Gi，访问模式 ReadWriteOnce。

用于 AlertManager 的三个 PV：大小 2Gi，访问模式 ReadWriteOnce。

（5） 日志配置持久化存储

日志的持久化主要是 ElasticSearch，需要提供块存储作持久化。我们在 infra01、infra02、infra03 节点分别额外配置了 100G 的裸磁盘，我将为 ElasticSearch 集群配置本地磁盘。在 master01 节点执行如下操作：

停止 fluentd Pod，停止日志采集

```
# oc project openshift-logging
# oc label node --all logging-infra-fluentd-
# for dc in $(oc get deploymentconfig --selector component=es -o name); do
    oc scale $dc --replicas=0
done
```

因为 ElasticSearch Pod 要配置宿主机的卷，需要开启特权运行

```
# oc adm policy add-scc-to-user privileged system:serviceaccount:openshift-logging:aggregated-logging-elasticsearch
# for dc in $(oc get deploymentconfig --selector component=es -o name); do
    oc patch $dc \
        -p '{"spec":{"template":{"spec":{"containers":[{"name":"elasticsearch","securityContext":{"privileged": true}}]}}}}'
done
```

为每个 infra 节点添加不同的标签

```
# oc label node infra01.example.cn es-node=infra01
# oc label node infra02.example.cn es-node=infra02
# oc label node infra03.example.cn es-node=infra03
```

获取所有 ElasticSearch 实例的 pod id

```
# oc get dc -l component=es
logging-es-data-master-a0dlefi1
logging-es-data-master-f3bzffyl
logging-es-data-master-zpgc6gfa
```

分别修改每个 ElasticSearch 的 nodeSelector，保证在每个节点运行一个 ElasticSearch 实

例。以第一个 es pod 为例，修改 nodeSelector 如下

oc edit dc logging-es-data-master-a0dlefi1

```
dnsPolicy: ClusterFirst
nodeSelector:
  es-node: infra01   #其他实例分别为 infra02、infra03
restartPolicy: Always
```

分别对 infra01、infra02、infra03 节点 100G 的裸磁盘进行格式化操作，并挂载到/es-data，设置开机自动挂载，设置目录/es-data 属主为 1000，操作过程略。对每个 ElasticSearch Pod 实例执行挂载 Volume。

```
# for dc in $(oc get deploymentconfig --selector component=es -o name); do
    oc set volume $dc \
        --add --overwrite --name=elasticsearch-storage \
        --type=hostPath --path=/es-data
done
```

启动 Pod 实例等待集群启动

```
# for dc in $(oc get deploymentconfig --selector component=es -o name); do
    oc rollout latest $dc
    oc scale $dc --replicas=1
done
```

启动 fluentd Pod，开始采集日志

```
# oc label node --all logging-infra-fluentd=true
```

到此为止，所有的安装工作就结束了，我们可以访问 https://master.example.com:8443 体验 OpenShift 了。