# 搭建本地yum源

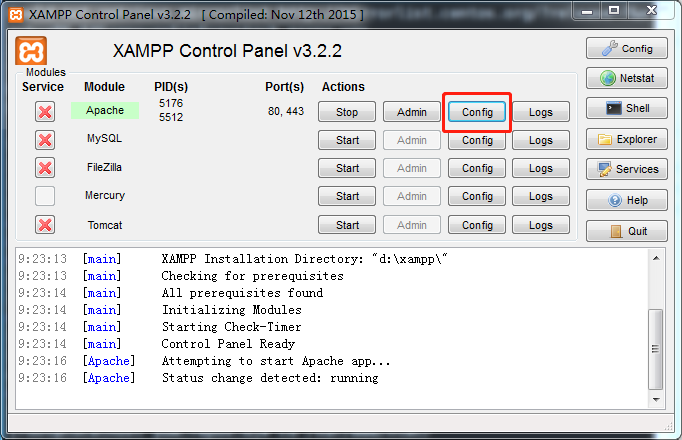
## centos系统

# yum install httpd

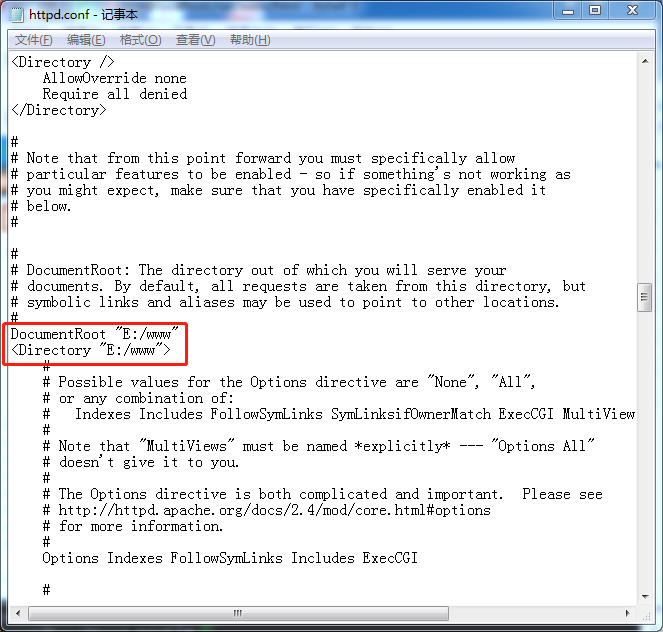
然后将文件复制到目录 /var/www/html中

## windows系统

安装xampp软件，打开Apache服务，将文件放在某个磁盘下的www文件夹下，



点击config下的Apache（http.conf）



改成存放文件的目录，打开Apache服务

关闭防火墙

# systemctl disable firewalld

# systemctl stop firewalld

在要装的服务器的/etc/yum.repos.d/下添加文件(IP:192.168.1.18)

把IP改为存有源文件的主机IP

yum7.repo

[centos]

name=centos7.2

baseurl= http://192.168.1.18/centos7.2/7.2/os/x86\_64/

enabled=1

gpgcheck=0

[epel]

name=epel7

baseurl= http://192.168.1.18/epel/7/x86\_64/

enabled=1

gpgcheck=0

mitaka.repo

[openstack]

name=mitaka

baseurl= http://192.168.1.18/openstack-mitaka/

enabled=1

gpgcheck=0

mariadb.repo

[mariadb]

name=mariadb10.1

baseurl= http://192.168.1.18/mariadb10.1/

enabled=1

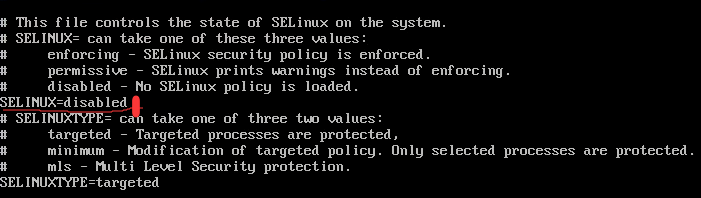
gpgcheck=0

# yum clean all

# yum makecache

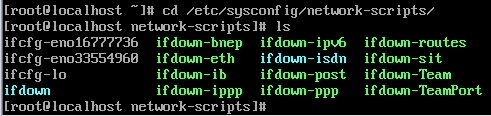
关闭selinux

# vi /etc/sysconfig/selinux



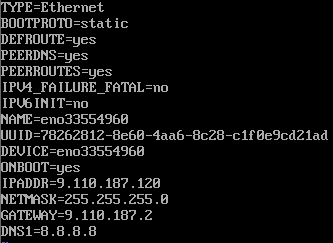
# 配置网络

# cd /etc/sysconfig/network-scripts/





（网卡1配置）



（网卡2配置）

TYPE=Ethernet

BOOTPROTO=static

DEFROUTE=yes

PEERDNS=yes

PEERROUTES=yes

IPV4\_FAILURE\_FATAL=no

IPV6INIT=no

NAME=enp4s0f1

UUID=b222b020-5999-48ad-9bbd-9fe0f64c55de

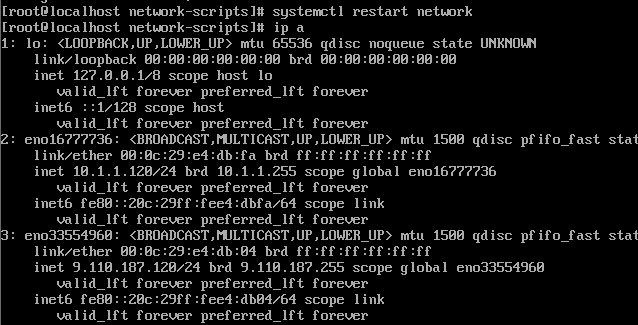
DEVICE=enp4s0f1

ONBOOT=yes

IPADDR=10.1.1.101

NETMASK=255.255.255.0

ip



# 更改主机名

# vi /etc/hostname

为配置的主机命名,暂命名为controller，保存后通过reboot重启后生效

配置域名解析，在所有节点的，/etc/hosts文件下添加以下内容（注意更改IP地址）

# controller

10.0.0.11 controller

# compute1

10.0.0.31 compute1

# block1

10.0.0.41 block1

# object1

10.0.0.51 object1

# object2

10.0.0.52 object2

验证连通性

在控制节点ping compute1

# ping -c 4 compute1

PING openstack.org (174.143.194.225) 56(84) bytes of data.

64 bytes from 174.143.194.225: icmp\_seq=1 ttl=54 time=18.3 ms

64 bytes from 174.143.194.225: icmp\_seq=2 ttl=54 time=17.5 ms

64 bytes from 174.143.194.225: icmp\_seq=3 ttl=54 time=17.5 ms

64 bytes from 174.143.194.225: icmp\_seq=4 ttl=54 time=17.4 ms

--- openstack.org ping statistics ---

4 packets transmitted, 4 received, 0% packet loss, time 3022ms

rtt min/avg/max/mdev = 17.489/17.715/18.346/0.364 ms

在其他节点ping controller，（多个节点互相ping一下）

**时**

其余节点

安装软件包：

# yum install chrony

编辑``/etc/chrony.conf`` 文件并注释除``server`` 值外的所有内容。修改它引用控制节点：

server controller（服务器IP地址） iburst

启动 NTP 服务并将其配置为随系统启动：

# systemctl enable chronyd.service

# systemctl start chronyd.service

验证操作

出现星号表示同步成功（联网情况下才会出现）

在其他节点上运行

# chronyc sources

210 Number of sources = 1

MS Name/IP address Stratum Poll Reach LastRx Last sample

===============================================================================

^\* controller 3 9 377 421 +15us[ -87us] +/- 15ms

# 安装OpenStack客户端

# yum install python-openstackclient

# yum install openstack-selinux

# nova服务

1. 安装软件包：
2. # yum install openstack-nova-compute
3. 编辑``/etc/nova/nova.conf``文件并完成下面的操作：

在``[DEFAULT]`` 和 [oslo\_messaging\_rabbit]部分，配置``RabbitMQ``消息队列的连接：

[DEFAULT]

rpc\_backend = rabbit

[oslo\_messaging\_rabbit]

rabbit\_host = controller

rabbit\_userid = openstack

rabbit\_password = RABBIT\_PASS

用你在 “RabbitMQ” 中为 “openstack” 选择的密码替换 “RABBIT\_PASS”。（实验室为 123456）

在 “[DEFAULT]” 和 “[keystone\_authtoken]” 部分，配置认证服务访问：

[DEFAULT]

auth\_strategy = keystone

[keystone\_authtoken]

...

auth\_uri = http://controller:5000

auth\_url = http://controller:35357

memcached\_servers = controller:11211

auth\_type = password

project\_domain\_name = default

user\_domain\_name = default

project\_name = service

username = nova

password = NOVA\_PASS

使用你在身份认证服务中设置的``nova`` 用户的密码替换``NOVA\_PASS``。（实验室为 123456）

注解

在 [keystone\_authtoken] 中注释或者删除其他选项。

在 [DEFAULT] 部分，配置 my\_ip 选项：

[DEFAULT]

...

my\_ip = MANAGEMENT\_INTERFACE\_IP\_ADDRESS

将其中的 MANAGEMENT\_INTERFACE\_IP\_ADDRESS 替换为计算节点上的管理网络接口的IP 地址，例如 :ref:[`](https://docs.openstack.org/mitaka/zh_CN/install-guide-rdo/nova-compute-install.html#id1)example architecture <overview-example-architectures>`中所示的第一个节点 10.0.0.31 。

* 在 [DEFAULT] 部分，使能 Networking 服务：

[DEFAULT]

...

use\_neutron = True

firewall\_driver = nova.virt.firewall.NoopFirewallDriver

在``[vnc]``部分，启用并配置远程控制台访问：

[vnc]

...

enabled = True

vncserver\_listen = 0.0.0.0

vncserver\_proxyclient\_address = $my\_ip

novncproxy\_base\_url = http://controller:6080/vnc\_auto.html

服务器组件监听所有的 IP 地址，而代理组件仅仅监听计算节点管理网络接口的 IP 地址。基本的 URL 指示您可以使用 web 浏览器访问位于该计算节点上实例的远程控制台的位置。

my\_ip = 192.168.1.22

注解

如果你运行浏览器的主机无法解析``controller`` 主机名，你可以将 ``controller``替换为你控制节点管理网络的IP地址。

* 在 [glance] 区域，配置镜像服务 API 的位置：

[glance]

...

api\_servers = http://controller:9292

* 在 [oslo\_concurrency] 部分，配置锁路径：

[ ]

...

lock\_path = /var/lib/nova/tmp

完成安装

1. 确定您的计算节点是否支持虚拟机的硬件加速。
2. $ egrep -c '(vmx|svm)' /proc/cpuinfo

如果这个命令返回了 one or greater 的值，那么你的计算节点支持硬件加速且不需要额外的配置。

如果这个命令返回了 zero 值，那么你的计算节点不支持硬件加速。你必须配置 libvirt 来使用 QEMU 去代替 KVM

* + 在 /etc/nova/nova.conf 文件的 [libvirt] 区域做出如下的编辑：

[libvirt]

...

virt\_type = qemu

1. 启动计算服务及其依赖，并将其配置为随系统自动启动：
2. # systemctl enable libvirtd.service openstack-nova-compute.service
3. # systemctl start libvirtd.service openstack-nova-compute.service

验证操作

注解

在控制节点上执行这些命令。

 获得 admin 凭证来获取只有管理员能执行的命令的访问权限：

$ source admin-openrc

 列出服务组件，以验证是否成功启动并注册了每个进程：

$ openstack compute service list

+----+--------------------+------------+----------+---------+-------+----------------------------+

| Id | Binary | Host | Zone | Status | State | Updated At |

+----+--------------------+------------+----------+---------+-------+----------------------------+

| 1 | nova-consoleauth | controller | internal | enabled | up | 2016-02-09T23:11:15.000000 |

| 2 | nova-scheduler | controller | internal | enabled | up | 2016-02-09T23:11:15.000000 |

| 3 | nova-conductor | controller | internal | enabled | up | 2016-02-09T23:11:16.000000 |

| 4 | nova-compute | compute1 | nova | enabled | up | 2016-02-09T23:11:20.000000 |

+----+--------------------+------------+----------+---------+-------+-----------

# neutron服务

**安装相关组件：**

**#** yum install openstack-neutron openstack-neutron-ml2 openstack-neutron-openvswitch

**配置各个组件：**

编辑 **/etc/neutron/neutron.conf**文件：

将[DEFAULT]中的关于连接数据库部分都注释掉！！！

[DEFAULT]

...

rpc\_backend = rabbit

auth\_strategy = keystone

[oslo\_messaging\_rabbit]

rabbit\_host = controller

rabbit\_userid = openstack

rabbit\_password = RABBIT\_PASS

[keystone\_authtoken]

...

auth\_uri = http://controller:5000

auth\_url = http://controller:35357

memcached\_servers = controller:11211

auth\_type = password

project\_domain\_name = default

user\_domain\_name = default

project\_name = service

username = neutron

password = NEUTRON\_PASS

[oslo\_concurrency]

...

lock\_path = /var/lib/neutron/tmp

替换**RABBIT\_PASS**为RABBITMQ的密码 （实验室为 123456）

替换**NEUTRON\_PASS**为用户neutron的密码 （实验室为 123456）

将[keystone\_authtoken]中的其他选项全部注释掉！！！

**配置其ovs agent：**

编辑 **/etc/neutron/plugins/ml2/openvswitch\_agent.ini**文件：

[ovs]

...

local\_ip = TUNNEL\_INTERFACE\_IP\_ADDRESS

[agent]

...

tunnel\_types = gre

l2\_population = true

[securitygroup]

...

firewall\_driver=neutron.agent.linux.iptables\_firewall.OVSHybridIptablesFirewallDriver

enable\_security\_group = true

替换**TUNNEL\_INTERFACE\_IP\_ADDRESS**为用于隧道网的网卡地址（租户网咯）

**配置nova,使其能使用网络服务：**

编辑 **/etc/nova/nova.conf**文件：

[neutron]

...

url = http://controller:9696

auth\_url = http://controller:35357

auth\_type = password

project\_domain\_name = default

user\_domain\_name = default

region\_name = RegionOne

project\_name = service

username = neutron

password = NEUTRON\_PASS

替换**NEUTRON\_PASS**为用户neutron的密码

**启用各项服务：**

**#** systemctl restart openstack-nova-compute.service

**#** systemctl enable neutron-openvswitch-agent.service

**#** systemctl start neutron-openvswitch-agent.service

**验证是否安装成功**

**控制节点执行下面操作**

source admin-openrc

$ neutron ext-list

+---------------------------+-----------------------------------------------+

| alias | name |

+---------------------------+-----------------------------------------------+

| default-subnetpools | Default Subnetpools |

| network-ip-availability | Network IP Availability |

| network\_availability\_zone | Network Availability Zone |

| auto-allocated-topology | Auto Allocated Topology Services |

| ext-gw-mode | Neutron L3 Configurable external gateway mode |

| binding | Port Binding |

| agent | agent |

| subnet\_allocation | Subnet Allocation |

| l3\_agent\_scheduler | L3 Agent Scheduler |

| tag | Tag support |

| external-net | Neutron external network |

| net-mtu | Network MTU |

| availability\_zone | Availability Zone |

| quotas | Quota management support |

| l3-ha | HA Router extension |

| provider | Provider Network |

| multi-provider | Multi Provider Network |

| address-scope | Address scope |

| extraroute | Neutron Extra Route |

| timestamp\_core | Time Stamp Fields addition **for** core resources |

| router | Neutron L3 Router |

| extra\_dhcp\_opt | Neutron Extra DHCP opts |

| dns-integration | DNS Integration |

| security-group | security-group |

| dhcp\_agent\_scheduler | DHCP Agent Scheduler |

| router\_availability\_zone | Router Availability Zone |

| rbac-policies | RBAC Policies |

| standard-attr-description | standard-attr-description |

| port-security | Port Security |

| allowed-address-pairs | Allowed Address Pairs |

| dvr | Distributed Virtual Router |

+---------------------------+-----------------------------------------------+

$ neutron agent-list

+--------------------------------------+--------------------+----------+-------------------+-------+----------------+---------------------------+

| id | agent\_type | host | availability\_zone | alive | admin\_state\_up | binary |

+--------------------------------------+--------------------+----------+-------------------+-------+----------------+---------------------------+

| 250ffcfd-afb1-43ed-b23b-77297cdf842b | L3 agent | network | nova | :-) | True | neutron-l3-agent |

| 2cc4f859-cf2f-4238-9053-210583ed96d5 | DHCP agent | network | nova | :-) | True | neutron-dhcp-agent |

| 7382d15a-8a75-405b-b829-748d5a93dd94 | Metadata agent | network | | :-) | True | neutron-metadata-agent |

| 8d504da9-5d70-4fd9-b8f6-5520fa7c7a5f | Open vSwitch agent | network | | :-) | True | neutron-openvswitch-agent |

| a09e5522-a4fc-4d21-be9a-968826386f3c | Open vSwitch agent | compute1 | | :-) | True | neutron-openvswitch-agent |

+--------------------------------------+--------------------+----------+----

# 集成docker

添加yum源（联网使用）

新建docker.repo内容为

[dockerrepo]

name=Docker Repository

baseurl=https://yum.dockerproject.org/repo/main/centos/7/

enabled=1

gpgcheck=1

gpgkey=https://yum.dockerproject.org/gpg

安装docker包

yum install docker-engine

查看已安装的docker软件

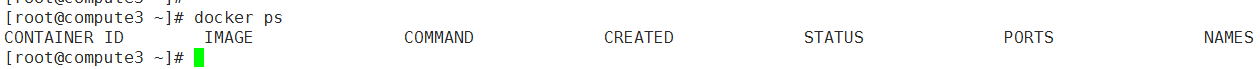
yum list installed | grep docker

启动docker服务

systemctl enable docker.service

systemctl start docker.service

安装完以后查看是否安装成功：



至此，docker安装成功。

usermod -aG docker nova

systemctl restart openstack-nova-compute

将nova-docker驱动上传到服务器

安装以下环境

yum install python-pip

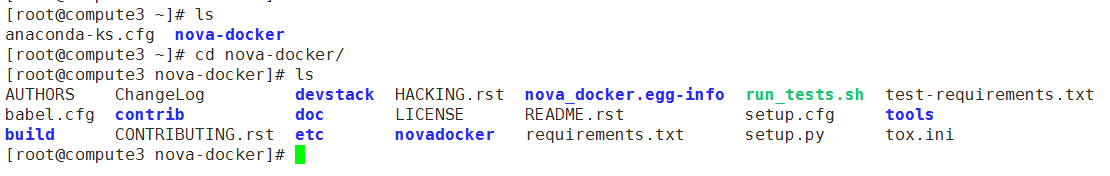
pip install --upgrade pip

yum install git

yum install gcc

解压nova-docker.tar.gz

进入nova-docker 目录



pip install docker //安装Python的docker库

pip install –r requirement.txt //安装依赖包

python setup.py install

至此，nova-docker安装成功。

在/etc/nova/rootwrap.d目录中创建docker.filters，也可以直接复制nova-docker中/etc/nova/中rootwrap.d的docker.filters，内容如下：

mkdir rootwrap.d

vi docker.filters

# nova-rootwrap command filters for setting up network in the docker driver

# This file should be owned by (and only-writeable by) the root user

[Filters]

# nova/virt/docker/driver.py: 'ln', '-sf', '/var/run/netns/.\*'

ln: CommandFilter, /bin/ln, root

然后改下拥有者：

chown -R root.nova /etc/nova/rootwrap.d

在计算节点节点中修改/etc/nova/nova.conf,添加对docker的驱动

[DEFAULT]

compute\_driver = novadocker.virt.docker.DockerDriver

# compute\_driver = libvirt.LibvirtDriver

**在控制节点的镜像配置中修改对docker镜像的支持**

在controller节点上修改/etc/glance/glance-api.conf,使之支持docker镜像

[DEFAULT]

container\_formats = ami,ari,aki,bare,ovf,docker

并重启镜像服务，包括api,registry

# systemctl restart openstack-glance-api.service openstack-glance-registry.service

**在计算节点重启计算服务，docker服务**

# systemctl restart libvirtd.service openstack-nova-compute.service

# systemctl restart docker.service

**然后在界面里面查看虚拟机管理器**

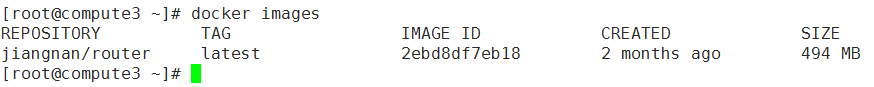


**可以看到compute3已经变成了docker类型**

在**计算节点**上传docker镜像

**镜像文件为jiangnan\_docker\_router.tar**

**上传方法：docker load --input jiangnan\_docker\_router.tar**



**下一步将该镜像文件复制到控制节点中，将该镜像上传到glance镜像服务中**

glance image-create --file jiangnan\_docker\_router.tar --container-format docker --disk-format raw --visibility public --name jiangnan/router

最后是启动容器，可以在界面中启动了，启动方法和虚拟机一致，注意可用域要填装了docker的计算节点。