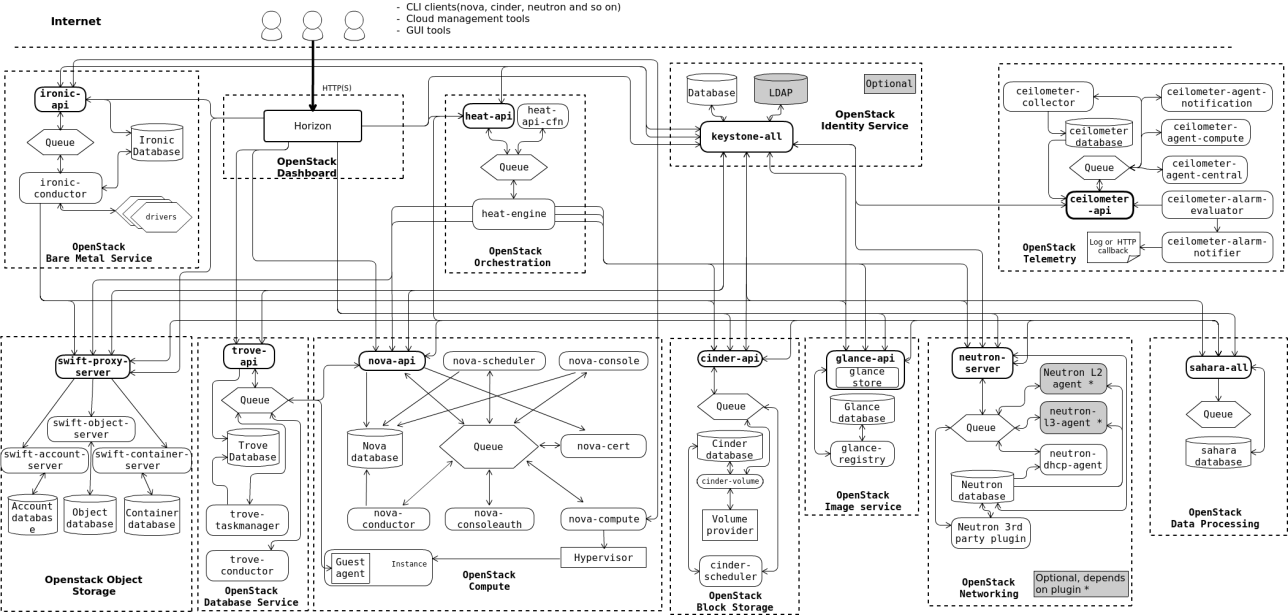
# Ch1 Logic Architecture



<https://docs.openstack.org/install-guide/get-started-logical-architecture.html>

## Services in Each Node

* **Controller Node**  
  **keystone service:** responsible for authentication, service rule and service token function.  
  **glance service:** manage image for VM;  
  **cinder-api:** responsible for block request;  
  **cinder-scheduler:** according to reservation policy, it will choose suitable cinder-volume node to deal with.  
  **nova-api:** it will provide RESTful api service;  
  **nova-scheduler:** it is used to choose suitable compute node to create VM;  
  **nova-conductor:** take charge of the visiting operation for nova-compute;  
  **nova-novncproxy:** provide the novncproxy agent for user to access VM;  
  **neutron-server:** take charge for user request and distribute task;  
  **neutron-linuxbridge-agent:** plugin for linux-bridge;  
  **neutron-l3-agent:** user create and manage virtual router;  
  **neutron-dhcp-service:** create and manage virtual dhcp server. Every virtual network has a dhcp service toprovide IP for VMs in this virtual network.  
  **neutron-metadata-agent:** every virtual network will create a neutron-ns-metadata-proxy process, this process will provide the metadata service for each VM.  
  neutron-lbass-agent: provide load balancing function.
* **Block Node**  
  cinder-volume: a storage space in storage node to manage storage device, every storage node will run a cinder-volume.
* **Compute Node**  
  nova-compute: provide compute service, used for interact with hypervision, control all VM operation and management.  
  neutron-linuxbridge-agent: plugin for bridge.

## Network Architecture



Interface1: communication network;

Interface2: management network;

Interface3: external network.

## Storage Architecture



sda: used for system installation;

sdb: used for storage resource when volume back-end is lvm in block node

sdc: used for ceph cluster installation

# Ch2 Basic Environment Establishment

## 2.1 Virtual Machine Establishment

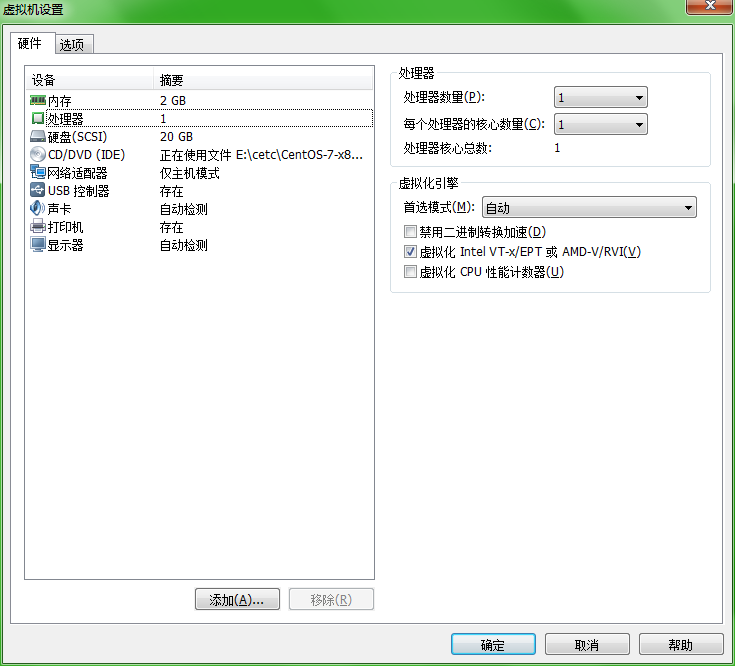
### 2.1.1 Establishment Step



Figure 2.1 Virtual Machine Establishment Procedure

### 2.1.2 Hardware Configuration in VMware

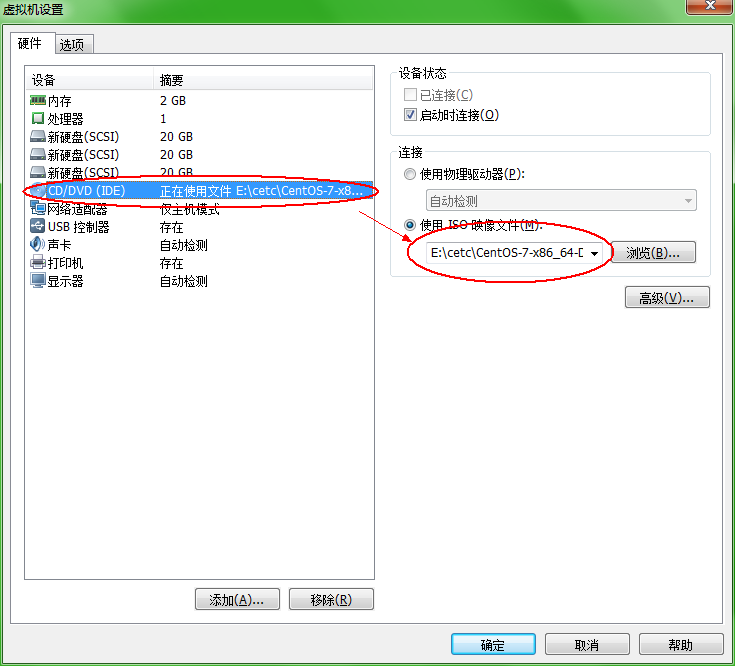
* Virtualization Settings



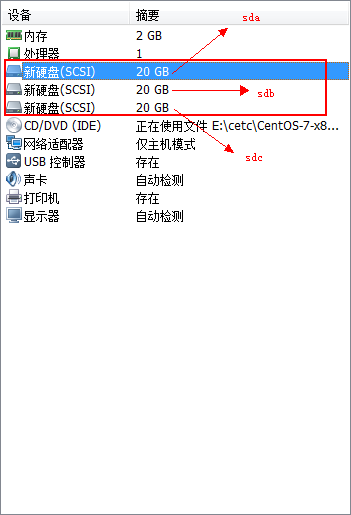
* Ram

One should set the ram size according to computer performance. CentOS can run with 512M.

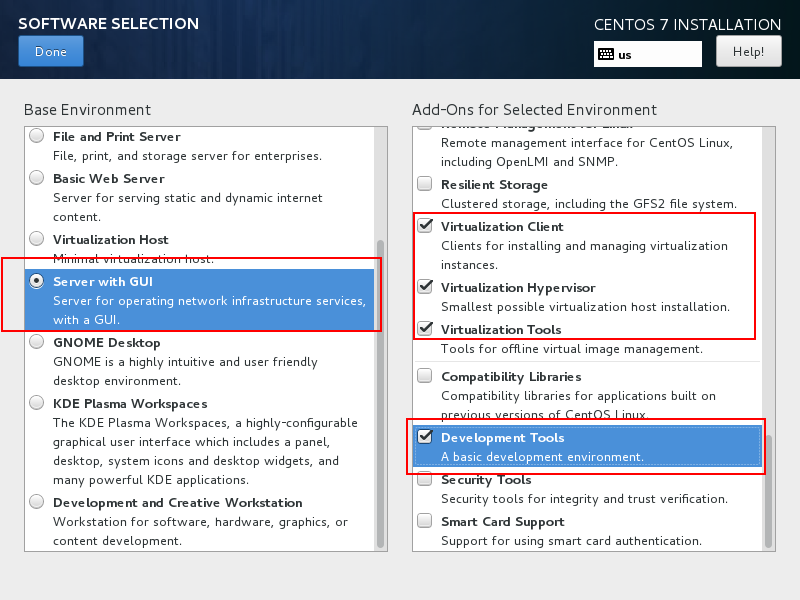
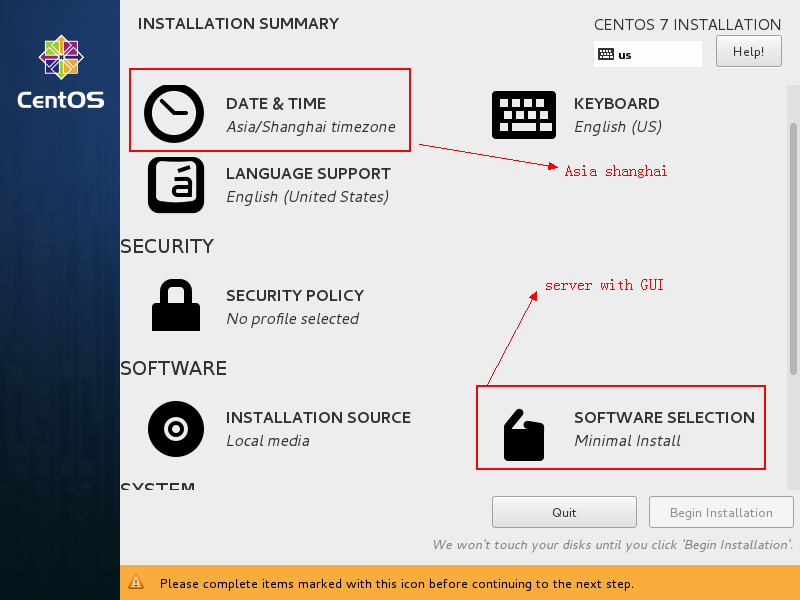
* cdrom



* virtual hard disk

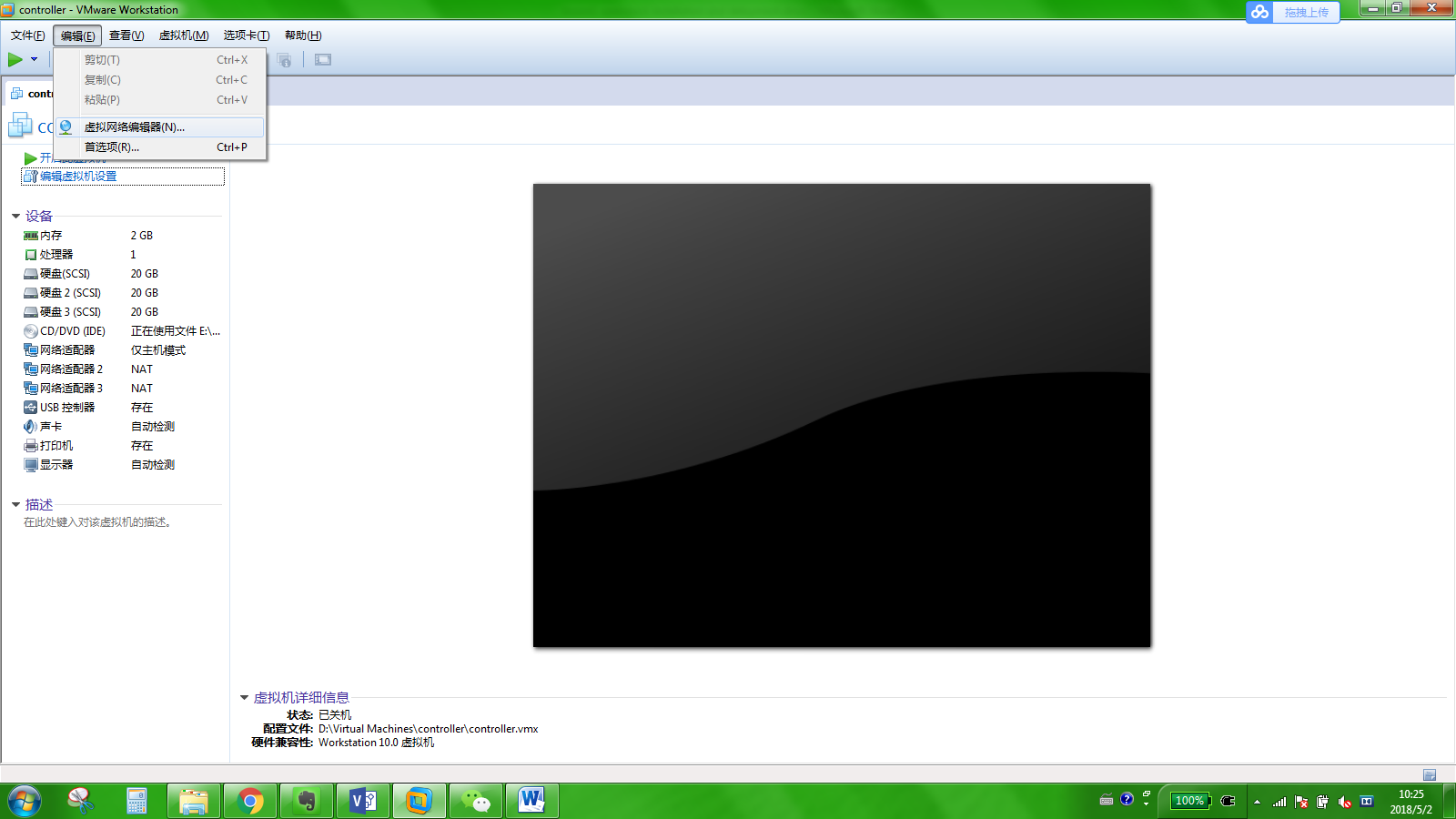
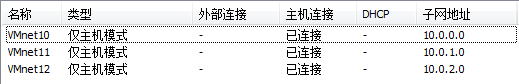


### 2.2.3 CentOS installation



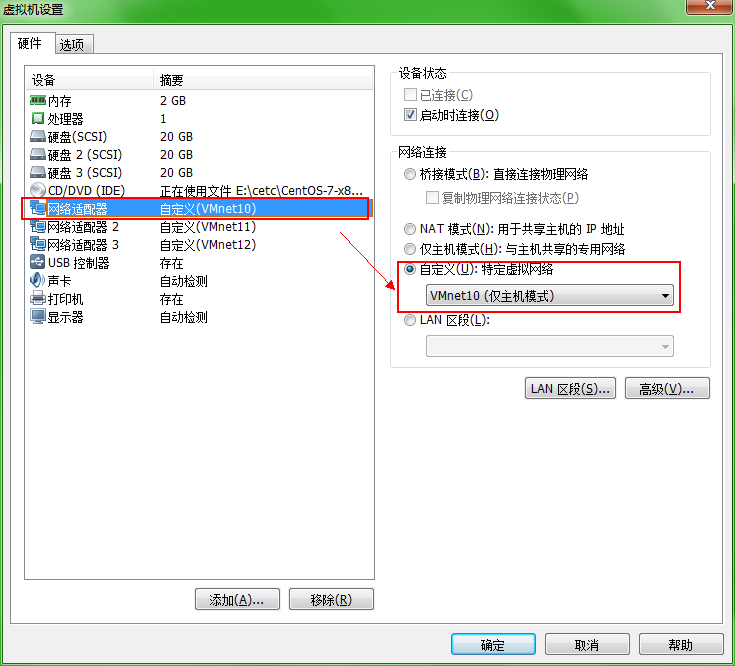
## 2.2 Network Hardware Configuration

### 2.2.1 Create Three Network

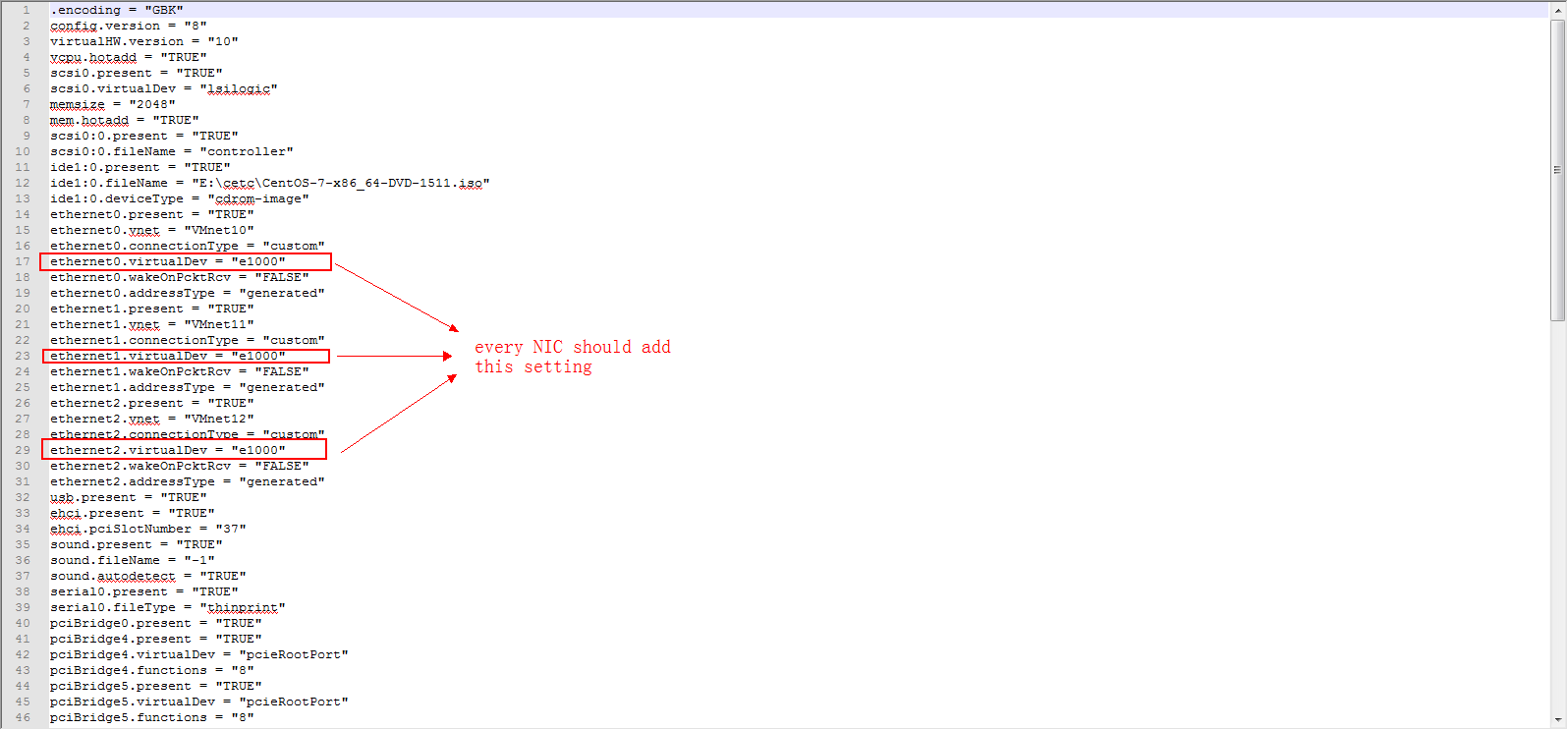
### 2.2.2 Create NIC

Controller Node: in this node, three NIC should be established, and each NIC connect to different network.



Tips: ifconfig –a commond cannot find the NIC

One should open \*.vmx file and add this command here.



### 2.2.3 VM Environment Configuration

In order to enable VM communicate with outside, selinux and firewall should close.

* **Close Selinux**

Security-Enhanced Linux (SELinux) is a [Linux kernel](https://en.wikipedia.org/wiki/Linux_kernel) [security module](https://en.wikipedia.org/wiki/Linux_Security_Modules) that provides a mechanism for supporting access control security policies, including [United States Department of Defense](https://en.wikipedia.org/wiki/United_States_Department_of_Defense)–style [mandatory access controls](https://en.wikipedia.org/wiki/Mandatory_access_control) (MAC).

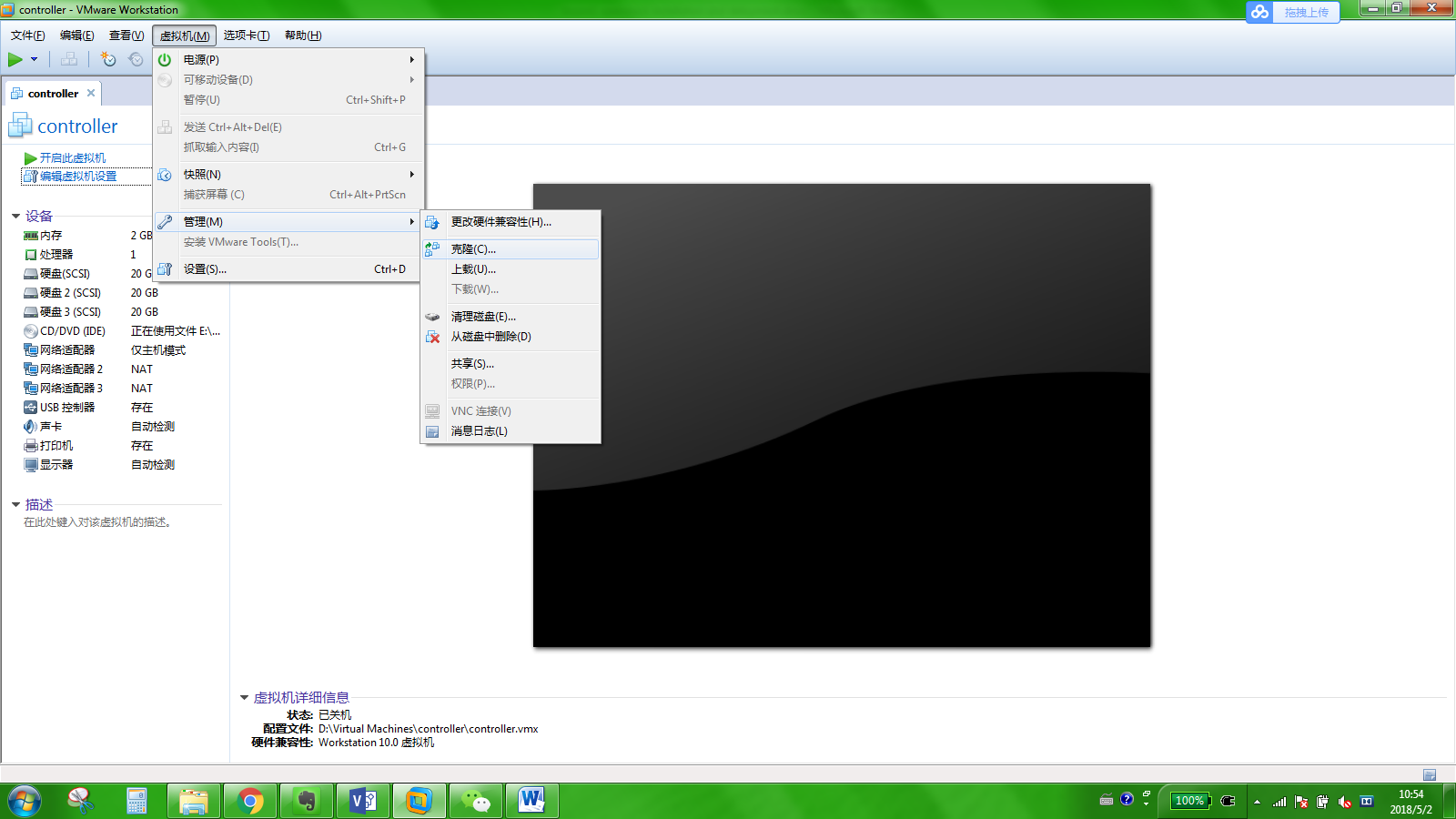
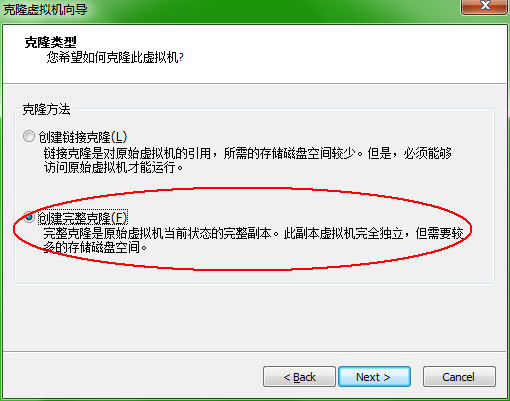
|  |
| --- |
| [root@localhost ~] vim /etc/selinux/config |

* **Close Firewall**

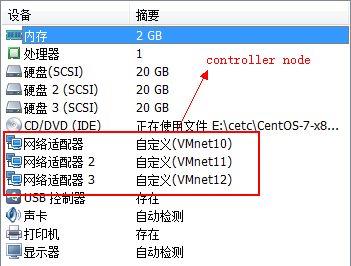
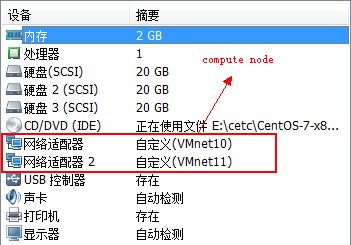
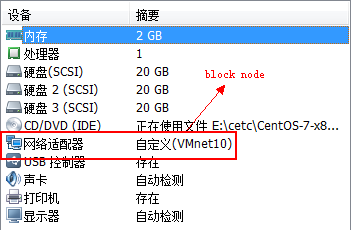
|  |
| --- |
| [root@localhost ~] systemctl status firewalld.service  [root@localhost ~] systemctl stop firewalld.service  [root@localhost ~] systemctl disable firewalld.service |

### 2.3.4 Configuration in Compute and Block Node

Compute and block node can be full clone from controller node.

Tips: It is important to select “create full clone”.

## 2.3 NIC Software Configuration

One should modify the highlight section in /etc/sysconfig/network-scripts/ifcfg-\*\*\*

|  |
| --- |
| TYPE=Ethernet  BOOTPROTO=none  DEFROUTE=yes  PEERDNS=yes  IPV4\_FAILURE\_FATAL=no  IPV6INIT=yes  IPV6\_AUTOCONF=yes  IPV6\_DEFROUTE=yes  IPV6\_PEERDNS=yes  IPV6=PEERROUTES=yes  IPV6\_FAILURE\_FATAL=no  NAME=eno1677746  UUID=1f4adb92-108d-44fc-822f-f15a09adc  DEVICE= eno1677746  ONBOOT=yes  IPADDR=10.0.0.11  PREFIX=24  GATEWAY=10.0.0.1 |

After those files have been configured, one should restart the network service to apply them.

|  |
| --- |
| [root@localhost~]#systemctl restart network.service |

Tip: one can install Xshell to connect to VM, so one can use it to do the next operations more convenient.

## 2.4 Software Installation

### 2.4.1 yum installation

## 3.5 Mariadb Database Configuration

**Step1**. Install mariadb on controller:

Yum install mariadb mariadb-server MySQL-python (tip: you should mount centos 7.2 to cdrom)

**Step2.** Configurate /etc/my.cnf.d/mariadb\_openstack.cnf file

Vim /etc/my.cnf.d/mariadb\_openstack.cnf

**Step3.** Add the following text into this file:

|  |
| --- |
| [mysqld]  bind-address =10.0.0.11  default-storage-engine =innodb  innodb\_file\_per\_table  collation-server=utf8\_general\_ci  init-connect=’SET NAMES utf8’  character-set-server = utf8 |

# Ch4 Keystone – identity Service

Keystone is one of the component of OpenStack. It is in charge of user security identification and permission control service.

## 4.1 Controller Node Installation and Configuration

## 4.2 Mariadb Database Installation

## 4.3 Identity Service Component Installation and Configuration

**Step1:** Install packages

|  |
| --- |
| Yum install openstack-keystone httpd mod\_wsgi python-openstackclient memcached python-memcached |

**Step2**: configure memcached service

|  |
| --- |
| Systemctl enable memcached.service  Systemctl start memcached.service |

**Step3:** configure and modify /etc/keystone/keystone.conf

|  |
| --- |
| [DEFAULT]  Admin\_token =ADMIN # a temporary token when we first configure keystone  verbose = true  debug = true  log\_file = keystone.log  log\_dir = /var/log/keystone  admin\_bind\_host= 10.0.0.11  public\_bind\_host=10.0.0.11  # connect keystone’s /usr/password/host/database  [database]  connection = mysql://keystone:own\_password@controller/keystone  [memcache]  servers=localhost:11211  # use uuid to verify and token to store memcache  [token]  provider= uuid  driver = memcache  # An implementation of the backend for persisting revocation events  [revoke]  driver = mysql |

**Step4**: synchronize database

|  |
| --- |
| su –s /bin/sh –c “keystone-manage db\_sync” keystone  # if synchronize succeed, keystone will establish tables  # No handlers could be found for logger “oslo\_config.cfg”, this warning can be ignored |

## 4.4 Configure Apache HTTP Server

Step1: modify /etc/httpd/conf/httpd.conf

vim /etc/httpd/conf/httpd.conf

|  |
| --- |
| # add those code into this file  ServerName controller |

Step2: copy wsgi-keystone.conf to /etc/httpd/conf.d directory

Step3:

|  |
| --- |
| Systemctl enable httpd.service  Systemctl start httpd.service  Systemctl status httpd.service |

|  |
| --- |
| 1. No such file or directory: AH02291: Cannot access directory ‘/var/log/apache2/’ for error log of vhost defined at /etc/httpd/conf.d/wsgi-keystone.conf:27 solution: <https://stackoverflow.com/questions/37796943/accidentally-deleted-var-log-apache2-and-now-cant-restart-apache> (answer2) |

## 4.5 Create Service Entity and API Endpoint

S1:

|  |
| --- |
| export OS\_TOKEN = ADMIN  export OS\_URL = <http://controller:35357/v3>  export OS\_IDENTITY\_API\_VERSION=3 |

S2: create service entity of identity service

|  |
| --- |
| openstack service create –name keystone –description “OpenStack Identity” identity |

|  |
| --- |
| The request you have made requires authentication. (Disable debug mode to suppress these detail.) (HTTP 401)   1. ADMIN\_TOKEN should be generated using "openssl rand -hex 10" command and should be applied in /etc/keystone/keystone.conf file "ADMIN\_TOKEN" section. ( eg : ADMIN\_TOKEN = < your generated ="" token="" id=""> 2. Then restart the service of httpd.service |

S3: create endpoint API of identity service

|  |
| --- |
| Openstack endpoint create –region RegionOne identity public <http://controller:5000/v2.0>  Openstack endpoint create –region RegioinOne identity internal <http://controller:5000/v2.0>  Openstack endpoint create –region RegionOne identity admin http://controller:5000/v2.0 |

## 4.6 Create projects, users and roles

4.6.1 Create an administrative project, user and role for administrative operations in your environment:

|  |
| --- |
| // 1. create an admin project  Openstack project create --domain default --description “Admin Project” admin  //2. create admin user with the password “admin”  Openstack user create --domain default --password-prompt admin  //3. create an admin role  Openstack role create admin  //4. Add admin role into admin project and admin user  Openstack role add --project admin --user admin admin |

4.6.2 This guidance uses a service that contains an unique user for each service that you add to your environment

|  |
| --- |
| //create a service project  Openstack project create --domain default --description “Service Project” service |

4.6.3 Examples

|  |
| --- |
| // create a demo project  Openstack project create --domain default --description “Demo Project” demo  //create demon user (password: demo)  Openstack user create --domain default --domain default --password-prompt demo  //create a role  Openstack role create user  // add demon role into demo project and demo user  Openstack role add --project demo –user demo user |

## 4.7 create openstack desktop environment script

|  |
| --- |
| //1. Edit admin-openrc.sh  Export OS\_PROJECT\_DOMAIN\_ID=default  Export OS\_USER\_DOMAIN\_ID=default  Export OS\_PROJECT\_NAME=admin  Export OS\_OSTENANT\_NAME=admin  Export OS\_USERNAME=admin  Export OS\_PASSWORD=admin  Export OS\_AUTH\_URL=http://controller:35357/v3  Export OS\_IDENTITY\_API\_VERSION=3  //2. Edit demo-openrc.sh file  Export OS\_PROJECT\_DOMAIN\_ID=default  Export OS\_USER\_DOMAIN\_ID=default  Export OS\_PROJECT\_NAME=demon  Export OS\_OSTENANT\_NAME=demon  Export OS\_USERNAME=demon  Export OS\_PASSWORD=demon  Export OS\_AUTH\_URL=http://controller:5000/v3  Export OS\_IDENTITY\_API\_VERSION=3 |

# Ch5 Glance – OpenStack Image Service

Glance is responsible for virtual machine image management service, include virtual machine image registration, login and retrieval service.

## 5.1 Controller Node Installation and Configuration

**Preparation**

Before one start to install and configure image service, a database, service credentials and API endpoints should be established first.

1. Create database  
   a) create database glance;  
   b) grant proper access to the glance database  
   grant all privileges on glance.\* to ‘glance’@’localhost’ identified by ‘glance’;  
   grant all privileges on glance.\* to ‘glance’@’%’ identified by ‘glance’;

|  |
| --- |
| Grant all privileges on DatabaseName.TableName to ‘user name’@’ip address’ identified by ‘password’ with grant option; |

1. Source the admin credentials to gain access to admin-only CLI commands;