# 龙芯云Openstack Wallaby 单机部署

## Openstack简介

OpenStack Wallaby 是 OpenStack 云计算平台的第 23 个版本，发布于 2021 年 4 月 14 日。OpenStack 是一个开源项目，旨在为公共和私有云提供大规模的云计算解决方案。Wallaby 版本引入了许多新功能、改进和修复，进一步提升了 OpenStack 的稳定性、安全性和性能。

OpenStack Wallaby 的主要功能和改进

1. 增强的安全性:

改进了 RBAC（基于角色的访问控制），提供了更细粒度的权限控制。

提高了对加密和密钥管理的支持，以确保数据在传输和存储时的安全性。

1. 简化的操作和管理:

增强了 Horizon（OpenStack 的 Web 界面）的功能，使管理员和用户能够更方便地管理资源。

改进了升级工具，使得从之前版本升级到 Wallaby 更加平滑。

1. 网络和 SDN（软件定义网络）改进:

Neutron 网络服务添加了对多个新网络驱动和插件的支持，增强了网络功能和可扩展性。

改进了对 IPv6 的支持，确保在现代网络环境中的兼容性。

1. 计算服务（Nova）增强:

添加了对新的硬件和虚拟化技术的支持，提升了计算资源的利用率。

改进了对混合云环境的支持，使得在不同云提供商之间的资源管理更加高效。

1. 存储服务（Cinder 和 Swift）更新:

Cinder 增强了对多个存储后端的支持，提供了更灵活的存储解决方案。

Swift 改进了对象存储的性能和可靠性，适应大规模存储需求。

1. 容器和 Kubernetes 支持:

Magnum 项目增强了对 Kubernetes 集群的管理，使得在 OpenStack 上运行容器变得更加简便和高效。

Zun 项目改进了对容器即服务的支持，提供了更多的容器运行时选项。

1. 边缘计算支持:

增强了对边缘计算场景的支持，使得 OpenStack 能够更好地在分布式和边缘环境中部署。

Wallaby 版本的主要组件

Nova: 计算服务，负责管理虚拟机实例。

Neutron: 网络服务，提供网络连接和管理功能。

Cinder: 块存储服务，提供持久化存储卷。

Glance: 镜像服务，管理虚拟机镜像。

Keystone: 认证服务，提供身份认证和权限管理。

Horizon: Web 界面，提供 OpenStack 的管理和使用界面。

Swift: 对象存储服务，提供高可用性对象存储。

Heat: 编排服务，管理云资源的编排和部署。

Magnum: 容器编排引擎，支持 Kubernetes、Docker Swarm 等。

Ironic: 裸机服务，管理裸机资源。

openEuler 24.03-LTS 版本官方LA源已经支持 OpenStack-Wallaby 版本，用户可以配置好 yum 源后根据此文档进行 OpenStack 部署。

## 环境描述

主机环境：OpenEuler 24.03-LTS 虚拟机

架构：Loongarch64

CPU： 8

CACHE： 8192Mbit

存储： 200 GBit

网络配置： bridge（桥接）模式 （桥接模式可以双向通信）

搭建模式：All In One模式（单机同时部署controller及compute节点）

**注意： 启动虚拟机的教程见《loongarch平台下使用virt-manager启动虚拟机》**

# 准备环境

本教程涉及的OpenStack配置插件：

KeyStone （Controller节点）

Glance 实例镜像管理工具（Controller节点）

Placement （Controller节点）

Nova （Compute节点）

horizon 桌面DashBoard（Controller节点）

## 配置yum源,安装edk2

1. 覆盖原有的/etc/yum.repos.d/openEuler.repo

[OS]

name=OS

baseurl=https://eulermaker.compass-ci.openeuler.openatom.cn/api/ems2/repositories/openEuler\_24.03\_LTS\_Epol\_Loongarch\_Multi-Version\_OpenStack\_Wallaby/openEuler%3A24.03-LTS/loongarch64/

metadata\_expire=1h

enabled=1

gpgcheck=0

priority=1

gpgkey=http://repo.openeuler.org/openEuler-24.03-LTS/OS//RPM-GPG-KEY-openEuler

[edk2]

name=edk2

#baseurl=https://eulermaker.compass-ci.openeuler.openatom.cn/api/ems3/repositories/openEuler-24.03-LTS-everything:loongarch/openEuler%3A24.03-LTS/loongarch64/

baseurl=https://eulermaker.compass-ci.openeuler.openatom.cn/api/ems3/repositories/openEuler-24.03-LTS-everything:loongarch-extra/openEuler%3A24.03-LTS/loongarch64/

metalink=https://mirrors.openeuler.org/metalink?repo=/everything&arch=

metadata\_expire=1h

enabled=1

gpgcheck=0

priority=20

gpgkey=http://repo.openeuler.org/openEuler-24.03-LTS/everything//RPM-GPG-KEY-openEuler

[EPOL]

name=EPOL

baseurl=https://eulermaker.compass-ci.openeuler.openatom.cn/api/ems2/repositories/openEuler-24.03-LTS-epol:loongarch/openEuler%3A24.03-LTS/loongarch64/

metadata\_expire=1h

enabled=1

gpgcheck=0

priority=30

gpgkey=http://repo.openeuler.org/openEuler-24.03-LTS/OS//RPM-GPG-KEY-openEuler

[everything]

name=everything

baseurl=https://eulermaker.compass-ci.openeuler.openatom.cn/api/ems3/repositories/openEuler-24.03-LTS-everything:loongarch/openEuler%3A24.03-LTS/loongarch64/

#baseurl=https://eulermaker.compass-ci.openeuler.openatom.cn/api/ems3/repositories/openEuler-24.03-LTS-everything:loongarch-extra/openEuler%3A24.03-LTS/loongarch64/

metalink=https://mirrors.openeuler.org/metalink?repo=/everything&arch=

metadata\_expire=1h

enabled=1

gpgcheck=0

priority=20

gpgkey=http://repo.openeuler.org/openEuler-24.03-LTS/everything//RPM-GPG-KEY-openEuler

[openstack]

name=openstack

baseurl=https://eulermaker.compass-ci.openeuler.openatom.cn/api/ems2/repositories/openEuler\_24.03\_LTS\_Epol\_Loongarch\_Multi-Version\_OpenStack\_Wallaby/openEuler%3A24.03-LTS/loongarch64/

metadata\_expire=1h

enabled=1

gpgcheck=0

priority=1

gpgkey=http://repo.openeuler.org/openEuler-24.03-LTS/OS//RPM-GPG-KEY-openEuler

1. 安装edk2
2. yum install edk2-devel.loongarch64 edk2-ovmf-loongarch64.noarch

## 修改主机名

将当前物理机名改为controller

1. hostnamectl set-hostname controller

修改/etc/hosts文件

1. cat >> /etc/hosts << EOF
2. 10.130.0.153 controller

**注意：10.130.0.153是当前物理机IP**

## 安装 SQL DataBase

执行如下命令，安装软件包。

1. yum install mariadb mariadb-server python3-PyMySQL

执行如下命令，创建并编辑 /etc/my.cnf.d/openstack.cnf 文件。

1. vim /etc/my.cnf.d/openstack.cnf
2. [mysqld]
3. bind-address = 10.130.0.153
4. default-storage-engine = innodb
5. innodb\_file\_per\_table = on
6. max\_connections = 4096
7. collation-server = utf8\_general\_ci
8. character-set-server = utf8

其中 bind-address 设置为控制节点的IP地址。

启动 DataBase 服务，并为其配置开机自启动：

1. systemctl enable mariadb.service && systemctl start mariadb.service

配置DataBase的默认密码（可选）

1. mysql\_secure\_installation

**注意：根据提示进行即可**

**注意：当前配置的密码与之后配置的密码最好进行记录或配置相同，以免后期使用忘记**

## 安装 RabbitMQ

执行如下命令，安装软件包。

1. yum install rabbitmq-server

启动 RabbitMQ 服务，并为其配置开机自启动。

1. systemctl enable rabbitmq-server.service && systemctl start rabbitmq-server.service

添加 OpenStack用户。

1. rabbitmqctl add\_user openstack PASS

**注意：替换 PASS，为 OpenStack 用户设置密码**

**声明：测试项目部署时当前PASS及后文包含PASS（区分大小写）的字段默认均设置为00000**

设置openstack用户权限，允许进行配置、写、读：

1. rabbitmqctl set\_permissions openstack ".\*" ".\*" ".\*"

## 2.5 安装 Memcached

执行如下命令，安装依赖软件包。

1. yum install memcached python3-memcached

编辑 /etc/sysconfig/memcached 文件。

1. vim /etc/sysconfig/memcached
2. OPTIONS="-l 127.0.0.1,::1,controller"

执行如下命令，启动 Memcached 服务，并为其配置开机启动。

1. systemctl enable memcached.service && systemctl start memcached.service

**注意：服务启动后，可以通过命令memcached-tool controller stats确保启动正常，服务可用，其中可以将controller替换为控制节点的IP地址。**

# 3. 安装 OpenStack

## 3.1 Keystone 安装

1. 创建 keystone 数据库并授权。
2. mysql -u root -p
3. MariaDB [(none)]> CREATE DATABASE keystone;
4. MariaDB [(none)]> GRANT ALL PRIVILEGES ON keystone.\* TO 'keystone'@'localhost' \
5. IDENTIFIED BY 'PASS';
6. MariaDB [(none)]> GRANT ALL PRIVILEGES ON keystone.\* TO 'keystone'@'%' \
7. IDENTIFIED BY 'PASS';
8. MariaDB [(none)]> exit

**注意：替换 PASS，为 Keystone 数据库设置密码**

1. 安装软件包。
2. yum install openstack-keystone httpd mod\_wsgi

配置keystone相关配置

1. vim /etc/keystone/keystone.conf
2. [database]
3. connection = mysql+pymysql://keystone:PASS@controller/keystone
4. [token]
5. provider = fernet

解释：

[database]部分，配置数据库入口

[token]部分，配置token provider

**注意：替换 PASS 为 Keystone 数据库的密码**

1. 同步数据库。
2. su -s /bin/sh -c "keystone-manage db\_sync" keystone

初始化Fernet密钥仓库。

1. keystone-manage fernet\_setup --keystone-user keystone --keystone-group keystone
2. keystone-manage credential\_setup --keystone-user keystone --keystone-group keystone
3. 启动服务
4. keystone-manage bootstrap --bootstrap-password PASS \
5. --bootstrap-admin-url http://controller:5000/v3/ \
6. --bootstrap-internal-url http://controller:5000/v3/ \
7. --bootstrap-public-url http://controller:5000/v3/ \
8. --bootstrap-region-id RegionOne

**注意：替换 PASS，为 admin 用户设置密码**

配置Apache HTTP server

1. vim /etc/httpd/conf/httpd.conf
2. ServerName controller
3. ln -s /usr/share/keystone/wsgi-keystone.conf /etc/httpd/conf.d/

解释

配置 ServerName 项引用控制节点

**注意：如果 ServerName 项不存在则需要创建**

1. 启动Apache HTTP服务。
2. systemctl enable httpd.service && systemctl start httpd.service
3. 创建环境变量配置。
4. cat << EOF >> ~/.admin-openrc
5. export OS\_PROJECT\_DOMAIN\_NAME=Default
6. export OS\_USER\_DOMAIN\_NAME=Default
7. export OS\_PROJECT\_NAME=admin
8. export OS\_USERNAME=admin
9. export OS\_PASSWORD=PASS
10. export OS\_AUTH\_URL=http:*//controller:5000/v3*
11. export OS\_IDENTITY\_API\_VERSION=3
12. export OS\_IMAGE\_API\_VERSION=2
13. EOF

**注意：替换 PASS 为 admin 用户的密码**

1. 依次创建domain, projects, users, roles，需要先安装好python3-openstackclient：
2. yum install python3-openstackclient

导入环境变量

1. source ~/.admin-openrc
2. 创建project service，其中 domain default 在 keystone-manage bootstrap 时已创建
3. openstack domain create *--description "An Example Domain" example*
4. openstack project create *--domain default --description "Service Project" service*
5. 创建（non-admin）project myproject，user myuser 和 role myrole，为 myproject 和 myuser 添加角色myrole
6. openstack project create *--domain default --description "Demo Project" myproject*
7. openstack user create *--domain default --password-prompt myuser*
8. openstack role create myrole
9. openstack role add *--project myproject --user myuser myrole*
10. 验证

取消临时环境变量OS\_AUTH\_URL和OS\_PASSWORD：

1. source ~/.admin-openrc
2. unset OS\_AUTH\_URL OS\_PASSWORD

为admin用户请求token：

1. openstack *--os-auth-url http://controller:5000/v3 \*
2. *--os-project-domain-name Default --os-user-domain-name Default \*
3. *--os-project-name admin --os-username admin token issue*

为myuser用户请求token：

1. openstack *--os-auth-url http://controller:5000/v3 \*
2. *--os-project-domain-name Default --os-user-domain-name Default \*
3. *--os-project-name myproject --os-username myuser token issue*

## 3.2 Glance 安装

1) 创建数据库、服务凭证和 API 端点

创建数据库：

1. mysql -u root -p
2. MariaDB [(none)]> CREATE DATABASE glance;
3. MariaDB [(none)]> GRANT ALL PRIVILEGES ON glance.\* TO 'glance'@'localhost' \
4. IDENTIFIED BY 'PASS';
5. MariaDB [(none)]> GRANT ALL PRIVILEGES ON glance.\* TO 'glance'@'%' \
6. IDENTIFIED BY 'PASS';
7. MariaDB [(none)]> exit

**注意:替换 PASS，为 glance 数据库设置密码**

创建服务凭证

1. source ~/.admin-openrc
2. openstack user create *--domain default --password-prompt glance*
3. openstack role add *--project service --user glance admin*
4. openstack service create *--name glance --description "OpenStack Image" image*

创建镜像服务API端点：

1. openstack endpoint create *--region RegionOne image public http://controller:9292*
2. openstack endpoint create *--region RegionOne image internal http://controller:9292*
3. openstack endpoint create *--region RegionOne image admin http://controller:9292*

安装软件包

1. yum install openstack-glance
2. 配置glance相关配置：
3. vim /etc/glance/glance-api.conf
4. [database]
5. connection = mysql+pymysql://glance:PASS@controller/glance
6. [keystone\_authtoken]
7. www\_authenticate\_uri  = http://controller:5000
8. auth\_url = http://controller:5000
9. memcached\_servers = controller:11211
10. auth\_type = password
11. project\_domain\_name = Default
12. user\_domain\_name = Default
13. project\_name = service
14. username = glance
15. password = PASS
16. [paste\_deploy]
17. flavor = keystone
18. [glance\_store]
19. stores = file,http
20. default\_store = file
21. filesystem\_store\_datadir = /var/lib/glance/images/

解释:

[database]部分，配置数据库入口

[keystone\_authtoken] [paste\_deploy]部分，配置身份认证服务入口

[glance\_store]部分，配置本地文件系统存储和镜像文件的位置

**注意：替换PASS 为 glance 数据库的密码和 glance 用户的密码**

1. 同步数据库：
2. su -s /bin/sh -c "glance-manage db\_sync" glance
3. 启动服务：
4. systemctl enable openstack-glance-api.service
5. systemctl start openstack-glance-api.service
6. 验证

下载镜像

1. source ~/.admin-openrc
2. wget https://repo.openeuler.org/openEuler-24.03-LTS/virtual\_machine\_img/loongarch64/openEuler-24.03-LTS-loongarch64.qcow2.xz && xz -d openEuler-24.03-LTS-loongarch64.qcow2.xz

向Image服务上传镜像：

1. openstack image create *--disk-format qcow2 --container-format bare \*
2. *--file openEuler-24.03-LTS-loongarch64.qcow2 --public openEuler-2403-LTS*

确认镜像上传并验证属性：

1. openstack image list

## 3.3 Placement安装

1. 创建数据库、服务凭证和 API 端点

创建数据库：

作为 root 用户访问数据库，创建 placement 数据库并授权。

1. mysql -u root -p
2. MariaDB [(none)]> CREATE DATABASE placement;
3. MariaDB [(none)]> GRANT ALL PRIVILEGES ON placement.\* TO 'placement'@'localhost' \
4. IDENTIFIED BY 'PASS';
5. MariaDB [(none)]> GRANT ALL PRIVILEGES ON placement.\* TO 'placement'@'%' \
6. IDENTIFIED BY 'PASS';
7. MariaDB [(none)]> exit

**注意替换 PASS 为 placement 数据库设置密码**

1. source ~/.admin-openrc

执行如下命令，创建 placement 服务凭证、创建 placement 用户以及添加‘admin’角色到用户‘placement’。

创建Placement API服务

1. openstack user create *--domain default --password-prompt placement*
2. openstack role add *--project service --user placement admin*
3. openstack service create *--name placement --description "Placement API" placement*

创建placement服务API端点：

1. openstack endpoint create *--region RegionOne placement public http://controller:8778*
2. openstack endpoint create *--region RegionOne placement internal http://controller:8778*
3. openstack endpoint create *--region RegionOne placement admin http://controller:8778*
4. 安装和配置

安装软件包：

1. yum install openstack-placement-api

配置placement：

编辑 /etc/placement/placement.conf 文件：

在[placement\_database]部分，配置数据库入口

在[api] [keystone\_authtoken]部分，配置身份认证服务入口

1. *# vim /etc/placement/placement.conf*
2. [placement\_database]
3. *# ...*
4. connection = mysql+pymysql://placement:PASS@controller/placement
5. [api]
6. *# ...*
7. auth\_strategy = keystone
8. [keystone\_authtoken]
9. *# ...*
10. auth\_url = http://controller:5000/v3
11. memcached\_servers = controller:11211
12. auth\_type = password
13. project\_domain\_name = Default
14. user\_domain\_name = Default
15. project\_name = service
16. username = placement
17. password = PASS

其中，替换 PASS 为 placement 数据库的密码和 placement 用户的密码。

同步数据库

su -s /bin/sh -c "placement-manage db sync" placement

1. systemctl restart httpd

启动httpd服务：

1. source ~/.admin-openrc
2. placement-status upgrade check
3. 验证

执行如下命令，执行状态检查：

1. yum install python3-osc-placement
2. openstack --os-placement-api-version 1.2 resource class list --sort-column name
3. openstack --os-placement-api-version 1.6 trait list --sort-column name

上述命令还安装了osc-placement，并列出可用的资源类别及特性：

## 3.4 Nova 安装

1. 创建数据库、服务凭证和 API 端点

创建数据库：

1. mysql -u root -p
2. MariaDB [(none)]> CREATE DATABASE nova\_api;
3. MariaDB [(none)]> CREATE DATABASE nova;
4. MariaDB [(none)]> CREATE DATABASE nova\_cell0;
5. MariaDB [(none)]> GRANT ALL PRIVILEGES ON nova\_api.\* TO 'nova'@'localhost' \
6. IDENTIFIED BY 'NOVA\_DBPASS';
7. MariaDB [(none)]> GRANT ALL PRIVILEGES ON nova\_api.\* TO 'nova'@'%' \
8. IDENTIFIED BY 'NOVA\_DBPASS';
9. MariaDB [(none)]> GRANT ALL PRIVILEGES ON nova.\* TO 'nova'@'localhost' \
10. IDENTIFIED BY 'NOVA\_DBPASS';
11. MariaDB [(none)]> GRANT ALL PRIVILEGES ON nova.\* TO 'nova'@'%' \
12. IDENTIFIED BY 'NOVA\_DBPASS';
13. MariaDB [(none)]> GRANT ALL PRIVILEGES ON nova\_cell0.\* TO 'nova'@'localhost' \
14. IDENTIFIED BY 'NOVA\_DBPASS';
15. MariaDB [(none)]> GRANT ALL PRIVILEGES ON nova\_cell0.\* TO 'nova'@'%' \
16. IDENTIFIED BY 'NOVA\_DBPASS';
17. MariaDB [(none)]> exit

**注意：替换NOVA\_DBPASS，为nova数据库设置密码**

1. source ~/.admin-openrc

创建nova服务凭证:

1. openstack user create *--domain default --password-prompt nova*
2. openstack role add *--project service --user nova admin*
3. openstack service create *--name nova --description "OpenStack Compute" compute*

创建nova API端点：

1. openstack endpoint create *--region RegionOne compute public http://controller:8774/v2.1*
2. openstack endpoint create *--region RegionOne compute internal http://controller:8774/v2.1*
3. openstack endpoint create *--region RegionOne compute admin http://controller:8774/v2.1*
4. 安装软件包
5. yum install openstack-nova-api openstack-nova-conductor \
6. openstack-nova-novncproxy openstack-nova-scheduler
7. yum install openstack-nova-compute
8. 配置nova相关配置
9. vim /etc/nova/nova.conf
10. [DEFAULT]
11. enabled\_apis = osapi\_compute,metadata
12. transport\_url = rabbit://openstack:RABBIT\_PASS@controller:5672/
13. my\_ip = 10.130.0.153
14. use\_neutron = true
15. log\_dir = /var/log/nova
16. firewall\_driver = nova.virt.firewall.NoopFirewallDriver
17. compute\_driver=libvirt.LibvirtDriver
18. instances\_path = /var/lib/nova/instances/
19. lock\_path = /var/lib/nova/tmp
20. [api\_database]
21. connection = mysql+pymysql://nova:NOVA\_DBPASS@controller/nova\_api
22. [database]
23. connection = mysql+pymysql://nova:NOVA\_DBPASS@controller/nova
24. [api]
25. auth\_strategy = keystone
26. [keystone\_authtoken]
27. www\_authenticate\_uri = http://controller:5000/
28. auth\_url = http://controller:5000/
29. memcached\_servers = controller:11211
30. auth\_type = password
31. project\_domain\_name = Default
32. user\_domain\_name = Default
33. project\_name = service
34. username = nova
35. password = NOVA\_PASS
36. [vnc]
37. enabled = true
38. server\_listen = $my\_ip
39. server\_proxyclient\_address = $my\_ip
40. novncproxy\_base\_url = http://10.130.0.153:6080/vnc\_auto.html
41. [libvirt]
42. virt\_type = qemu
43. cpu\_mode = custom
44. cpu\_model = la464
45. [glance]
46. api\_servers = http://controller:9292
47. [oslo\_concurrency]
48. lock\_path = /var/lib/nova/tmp
49. [placement]
50. region\_name = RegionOne
51. project\_domain\_name = Default
52. project\_name = service
53. auth\_type = password
54. user\_domain\_name = Default
55. auth\_url = http://controller:5000/v3
56. username = placement
57. password = PLACEMENT\_PASS
58. [neutron]
59. auth\_url = http://controller:5000
60. auth\_type = password
61. project\_domain\_name = default
62. user\_domain\_name = default
63. region\_name = RegionOne
64. project\_name = service
65. username = neutron
66. password = NEUTRON\_PASS
67. service\_metadata\_proxy = true
68. metadata\_proxy\_shared\_secret = METADATA\_SECRET

解释

[default]部分，启用计算和元数据的API，配置RabbitMQ消息队列入口，配置my\_ip，启用网络服务neutron；

[api\_database] [database]部分，配置数据库入口；

[api] [keystone\_authtoken]部分，配置身份认证服务入口；

[vnc]部分，启用并配置远程控制台入口；

[glance]部分，配置镜像服务API的地址；

[oslo\_concurrency]部分，配置lock path；

[placement]部分，配置placement服务的入口。

**注意**

**替换 RABBIT\_PASS 为 RabbitMQ 中 openstack 账户的密码；**

**配置 my\_ip 为控制节点的管理IP地址；**

**替换 NOVA\_DBPASS 为nova数据库的密码；**

**替换 NOVA\_PASS 为nova用户的密码；**

**替换 PLACEMENT\_PASS 为placement用户的密码；**

**替换 NEUTRON\_PASS 为neutron用户的密码；**

**替换METADATA\_SECRET为合适的元数据代理secret（00000与PASS设置一致）。**

1. 同步数据库
2. 同步nova-api数据库：
3. su -s /bin/sh -c "nova-manage api\_db sync" nova
4. 注册cell0数据库：
5. su -s /bin/sh -c "nova-manage cell\_v2 map\_cell0" nova
6. 创建cell1 cell：
7. su -s /bin/sh -c "nova-manage cell\_v2 create\_cell --name=cell1 --verbose" nova
8. 同步nova数据库：
9. su -s /bin/sh -c "nova-manage db sync" nova
10. 验证cell0和cell1注册正确：
11. su -s /bin/sh -c "nova-manage cell\_v2 list\_cells" nova
12. 添加计算节点到openstack集群
13. su -s /bin/sh -c "nova-manage cell\_v2 discover\_hosts --verbose" nova

1. 启动服务
2. systemctl enable \
3. openstack-nova-api.service \
4. openstack-nova-scheduler.service \
5. openstack-nova-conductor.service \
6. openstack-nova-novncproxy.service
7. systemctl start \
8. openstack-nova-api.service \
9. openstack-nova-scheduler.service \
10. openstack-nova-conductor.service \
11. openstack-nova-novncproxy.service
12. systemctl enable libvirtd.service openstack-nova-compute.service
13. systemctl start libvirtd.service openstack-nova-compute.service
14. 验证
15. source ~/.admin-openrc
16. 列出服务组件，验证每个流程都成功启动和注册：
17. openstack compute service list
18. 列出身份服务中的API端点，验证与身份服务的连接：
19. openstack catalog list
20. 列出镜像服务中的镜像，验证与镜像服务的连接：
21. openstack image list
22. 检查cells是否运作成功，以及其他必要条件是否已具备。
23. nova-status upgrade check

## 3.5 Neutron 安装

1. 创建数据库、服务凭证和 API 端点

创建数据库：

1. mysql -u root -p
2. MariaDB [(none)]> CREATE DATABASE neutron;
3. MariaDB [(none)]> GRANT ALL PRIVILEGES ON neutron.\* TO 'neutron'@'localhost' \
4. IDENTIFIED BY 'NEUTRON\_DBPASS';
5. MariaDB [(none)]> GRANT ALL PRIVILEGES ON neutron.\* TO 'neutron'@'%' \
6. IDENTIFIED BY 'NEUTRON\_DBPASS';
7. MariaDB [(none)]> exit

**注意:替换 NEUTRON\_DBPASS 为 neutron 数据库设置密码。**

source ~/.admin-openrc

创建neutron服务凭证

1. openstack user create *--domain default --password-prompt neutron*
2. openstack role add *--project service --user neutron admin*
3. openstack service create *--name neutron --description "OpenStack Networking" network*

创建Neutron服务API端点：

1. openstack endpoint create *--region RegionOne network public http://controller:9696*
2. openstack endpoint create *--region RegionOne network internal http://controller:9696*
3. openstack endpoint create *--region RegionOne network admin http://controller:9696*
4. 安装软件包：
5. yum install openstack-neutron openstack-neutron-linuxbridge ebtables ipset \
6. openstack-neutron-ml2
7. yum install openstack-neutron-linuxbridge ebtables ipset
8. 配置neutron相关配置：

配置主体配置

1. vim /etc/neutron/neutron.conf
2. [database]
3. connection = mysql+pymysql://neutron:NEUTRON\_DBPASS@controller/neutron
4. [DEFAULT]
5. core\_plugin = ml2
6. service\_plugins = router
7. allow\_overlapping\_ips = true
8. transport\_url = rabbit://openstack:RABBIT\_PASS@controller
9. auth\_strategy = keystone
10. notify\_nova\_on\_port\_status\_changes = true
11. notify\_nova\_on\_port\_data\_changes = true
12. api\_workers = 3
13. [keystone\_authtoken]
14. www\_authenticate\_uri = http://controller:5000
15. auth\_url = http://controller:5000
16. memcached\_servers = controller:11211
17. auth\_type = password
18. project\_domain\_name = Default
19. user\_domain\_name = Default
20. project\_name = service
21. username = neutron
22. password = NEUTRON\_PASS
23. [nova]
24. auth\_url = http://controller:5000
25. auth\_type = password
26. project\_domain\_name = Default
27. user\_domain\_name = Default
28. region\_name = RegionOne
29. project\_name = service
30. username = nova
31. password = NOVA\_PASS
32. [oslo\_concurrency]
33. lock\_path = /var/lib/neutron/tmp

解释

[database]部分，配置数据库入口；

[default]部分，启用ml2插件和router插件，允许ip地址重叠，配置RabbitMQ消息队列入口；

[default] [keystone]部分，配置身份认证服务入口；

[default] [nova]部分，配置网络来通知计算网络拓扑的变化；

[oslo\_concurrency]部分，配置lock path。

**注意**

**替换NEUTRON\_DBPASS为 neutron 数据库的密码；**

**替换RABBIT\_PASS为 RabbitMQ中openstack 账户的密码；**

**替换NEUTRON\_PASS为 neutron 用户的密码；**

**替换NOVA\_PASS为 nova 用户的密码。**

1. 配置ML2插件：
2. vim /etc/neutron/plugins/ml2/ml2\_conf.ini
3. [ml2]
4. type\_drivers = flat,vlan,vxlan
5. tenant\_network\_types = vxlan
6. mechanism\_drivers = linuxbridge,l2population
7. extension\_drivers = port\_security
8. [ml2\_type\_flat]
9. flat\_networks = provider
10. [ml2\_type\_vxlan]
11. vni\_ranges = 1:1000
12. [securitygroup]
13. enable\_ipset = true
14. 创建/etc/neutron/plugin.ini的符号链接
15. ln -s /etc/neutron/plugins/ml2/ml2\_conf.ini /etc/neutron/plugin.ini

**注意**

**[ml2]部分，启用 flat、vlan、vxlan 网络，启用 linuxbridge 及 l2population 机制，启用端口安全扩展驱动；**

**[ml2\_type\_flat]部分，配置 flat 网络为 provider 虚拟网络；**

**[ml2\_type\_vxlan]部分，配置 VXLAN 网络标识符范围；**

**[securitygroup]部分，配置允许 ipset。**

**补充**

**l2 的具体配置可以根据用户需求自行修改，本文使用的是provider network + linuxbridge**

1. 配置 Linux bridge 代理：
2. vim /etc/neutron/plugins/ml2/linuxbridge\_agent.ini
3. [linux\_bridge]
4. physical\_interface\_mappings = provider:PROVIDER\_INTERFACE\_NAME
5. [vxlan]
6. enable\_vxlan = true
7. local\_ip = OVERLAY\_INTERFACE\_IP\_ADDRESS
8. l2\_population = true
9. [securitygroup]
10. enable\_security\_group = true
11. firewall\_driver = neutron.agent.linux.iptables\_firewall.IptablesFirewallDriver

解释

[linux\_bridge]部分，映射 provider 虚拟网络到物理网络接口(网卡名称)；

[vxlan]部分，启用 vxlan 覆盖网络，配置处理覆盖网络的物理网络接口 IP 地址，启用 layer-2 population；

[securitygroup]部分，允许安全组，配置 linux bridge iptables 防火墙驱动。

**注意**

**替换PROVIDER\_INTERFACE\_NAME为物理网络接口；**

**替换OVERLAY\_INTERFACE\_IP\_ADDRESS为控制节点的IP地址。**

6） 配置Layer-3代理：

1. vim /etc/neutron/l3\_agent.ini
2. [DEFAULT]
3. interface\_driver = linuxbridge

解释

在[default]部分，配置接口驱动为linuxbridge

1. 配置DHCP代理：
2. vim /etc/neutron/dhcp\_agent.ini
3. [DEFAULT]
4. interface\_driver = linuxbridge
5. dhcp\_driver = neutron.agent.linux.dhcp.Dnsmasq
6. enable\_isolated\_metadata = true

解释

[default]部分，配置linuxbridge接口驱动、Dnsmasq DHCP驱动，启用隔离的元数据。

1. 配置metadata代理：
2. vim /etc/neutron/metadata\_agent.ini
3. [DEFAULT]
4. nova\_metadata\_host = controller
5. metadata\_proxy\_shared\_secret = METADATA\_SECRET

解释

[default]部分，配置元数据主机和shared secret。

**注意**

**替换METADATA\_SECRET为合适的元数据代理secret。**

1. 配置nova相关配置
2. vim /etc/nova/nova.conf
3. [neutron]
4. auth\_url = http://controller:5000
5. auth\_type = password
6. project\_domain\_name = default
7. user\_domain\_name = default
8. region\_name = RegionOne
9. project\_name = service
10. username = neutron
11. password = NEUTRON\_PASS
12. service\_metadata\_proxy = true
13. metadata\_proxy\_shared\_secret = METADATA\_SECRET

**解释：[neutron]部分，配置访问参数，启用元数据代理，配置secret。**

**注意**

**替换NEUTRON\_PASS为 neutron 用户的密码；**

**替换METADATA\_SECRET为合适的元数据代理secret。**

1. 同步数据库：
2. su -s /bin/sh -c "neutron-db-manage --config-file /etc/neutron/neutron.conf \
3. --config-file /etc/neutron/plugins/ml2/ml2\_conf.ini upgrade head" neutron
4. 重启计算API服务：
5. systemctl restart openstack-nova-api.service
6. 启动网络服务
7. systemctl enable neutron-server.service neutron-linuxbridge-agent.service \
8. neutron-dhcp-agent.service neutron-metadata-agent.service
9. systemctl enable neutron-l3-agent.service
10. systemctl restart openstack-nova-api.service neutron-server.service
11. neutron-linuxbridge-agent.service neutron-dhcp-agent.service \
12. neutron-metadata-agent.service neutron-l3-agent.service
13. systemctl enable neutron-linuxbridge-agent.service
14. systemctl restart neutron-linuxbridge-agent.service openstack-nova-compute.service
15. 验证

验证 neutron 代理启动成功：

1. openstack network agent list

## 3.6 Horizon 安装

1. 安装软件包
2. yum install openstack-dashboard
3. 修改文件

修改变量

1. vim /etc/openstack-dashboard/local\_settings
2. OPENSTACK\_HOST = "controller"
3. ALLOWED\_HOSTS = ['\*', ]
4. SESSION\_ENGINE = 'django.contrib.sessions.backends.cache'
5. CACHES = {
6. 'default': {
7. 'BACKEND': 'django.core.cache.backends.memcached.MemcachedCache',
8. 'LOCATION': 'controller:11211',
9. }
10. }
11. OPENSTACK\_KEYSTONE\_URL = "http://controller:5000/v3"
12. OPENSTACK\_KEYSTONE\_MULTIDOMAIN\_SUPPORT = True
13. OPENSTACK\_KEYSTONE\_DEFAULT\_DOMAIN = "Default"
14. OPENSTACK\_KEYSTONE\_DEFAULT\_ROLE = "member"
15. WEBROOT = '/dashboard'
16. POLICY\_FILES\_PATH = "/etc/openstack-dashboard"
17. OPENSTACK\_API\_VERSIONS = {
18. "identity": 3,
19. "image": 2,
20. "volume": 3,
21. }

重启 httpd 服务

1. systemctl restart httpd.service memcached.service
2. 验证

打开浏览器，输入网址http://HOSTIP/dashboard/，登录 horizon。

**注意：替换HOSTIP为控制节点IP地址**

## 3.7 Tempest 安装

# 4. 部署虚拟机

经过上述步骤，Openstack已经成功部署，本节进行qcow虚拟机的部署启动

## 4.1 镜像安装

1. 镜像下载

选择两个qcow2镜像进行部署，分别是archlinux和openEuler系统，获取地址：

<https://repo.openeuler.org/openEuler-24.03-LTS/virtual_machine_img/loongarch64/openEuler-24.03-LTS-loongarch64.qcow2.xz>（需要解压）

<https://github.com/zunley/archlinux-loongarch64/releases/download/v0.0.1/archlinux.qcow2>

1. 镜像部署

部署命令

openstack image create --disk-format qcow2 --container-format bare --file openEuler-24.03-LTS-loongarch64.qcow2 --public cirros

查看命令

openstack image list

## 4.2 实例创建

在网页进行创建，