Openstack Liberty is an older version of Openstack that has been deprecated and it requires an older version of OS repository.

Steps:

Ubuntu Server 14.04 LTS was installed on stack9 along with the proper network configurations.
These commands were ran to grab the necessary packages:

```
apt-get install software-properties-common
add-apt-repository cloud-archive:liberty
apt-get update && apt-get dist-upgrade
```

Install Openstack Client:

```
apt-get install python-openstackclient
```

Edit hosts file:

```
Vi /etc/hosts
```

Add the following line to the file:

```
192.168.151.9        controller      stack9  stack9.myhu.cloud
```

Openstack relies heavily on its sql database for the management of its modules and the operating of the cloud.

Install SQL Database (no password was set):

```
apt-get install mariadb-server python-pymysql
```

Edit openstack sql config file:

```
vi /etc/mysql/conf.d/mysqld_openstack.cnf
```

Add to the file:

```
[mysqld]
bind-address = 192.168.151.9
default-storage-engine = innodb
Innodb_file_per_table
collation-server = utf8_general_ci
init-connect = 'SET NAMES utf8'
character-set-server = utf8
```

Restart the database server:

```
service mysql restart
```

Secure the database server (no password was set):

```
Mysql_secure_installation
```

The Openstack Telemetry service uses a NoSQL database to store information.

```
apt-get install mongodb-server mongodb-clients python-pymongo
```

Edit the /etc/mongodb.conf

```
bind_ip = 192.168.151.9
smallfiles = true
```

Restart the service:

```
service mongodb restart
```

RabbitMQ is installed for the usage of messaging queue services:

```
apt-get install rabbitmq-server
```

Create a user for openstack:

```
rabbitmqctl add_user openstack P@ssw0rd
```

Set permission for the user:

```
rabbitmqctl set_permissions openstack ".*" ".*" ".*"
```

Keystone Configuration:

Create a database for keystone:

```
mysql -u root -p
CREATE DATABASE keystone;
GRANT ALL PRIVILEGES ON keystone.* TO 'keystone'@'localhost' IDENTIFIED BY ‘P@ssw0rd';
GRANT ALL PRIVILEGES ON keystone.* TO 'keystone'@'%' IDENTIFIED BY 'P@ssw0rd';
Exit
```

Quick check:

```
mysql -h localhost -P 1377 -D keystone -u keystone -pP@ssw0rd
```

Disable the keystone service from starting automatically after installation:

```
echo "manual" > /etc/init/keystone.override
```

Install HTTPD server and other necessary packages:

```
apt-get install keystone apache2 libapache2-mod-wsgi memcached python-memcache
```

Generate a random value for the keystone admin token:

```
openssl rand -hex 10
9cad17371f842705310e
```

Edit /etc/keystone/keystone.conf

```
[DEFAULT]
admin_token = 406421e15156cd3bbd43
[database]
connection =mysql+pymysql://keystone:P@ssw0rd@controller/keystone
[memcache]
servers = localhost:11211
[token]
provider = uuid
driver = memcache
[revoke]
driver = sql
```

Comment out these lines

```
#sqlite_db = oslo.sqlite
#sqlite_synchronous = true
#backend = sqlalchemy
```

Populate the keystone database:

```
su -s /bin/sh -c "keystone-manage db_sync" keystone
```

Configure Apache2:

Edit /etc/apache2/apache2.conf and add the following line:

```
ServerName controller
```

Create the wsgi-keystone.conf file:

```
vi /etc/apache2/sites-available/wsgi-keystone.conf
```

Add the following to the file:

```
Listen 5000
Listen 35357

<VirtualHost *:5000>
    WSGIDaemonProcess keystone-public processes=5 threads=1 user=keystone
group=keystone display-name=%{GROUP}
    WSGIProcessGroup keystone-public
    WSGIScriptAlias / /usr/bin/keystone-wsgi-public
    WSGIApplicationGroup %{GLOBAL}
    WSGIPassAuthorization On
    <IfVersion >= 2.4>
       ErrorLogFormat"%{cu}t %M"
    </IfVersion>
     ErrorLog /var/log/apache2/keystone.log
     CustomLog /var/log/apache2/keystone_access.log combined
    <Directory /usr/bin>
       <IfVersion >= 2.4>
          Require all granted
       </IfVersion>
       <IfVersion < 2.4>
          Order allow,deny
          Allow from all
       </IfVersion>
     </Directory>
</VirtualHost>


<VirtualHost *:35357>
    WSGIDaemonProcess keystone-admin processes=5 threads=1 user=keystone
group=keystone display-name=%{GROUP}
    WSGIProcessGroup keystone-admin
    WSGIScriptAlias / /usr/bin/keystone-wsgi-admin
    WSGIApplicationGroup %{GLOBAL}
    WSGIPassAuthorization On
    <IfVersion >= 2.4>
      ErrorLogFormat "%{cu}t %M"
    </IfVersion>
    ErrorLog /var/log/apache2/keystone.log
    CustomLog /var/log/apache2/keystone_access.log combined
   <Directory /usr/bin>
     <IfVersion >= 2.4>
       Require all granted
     </IfVersion>
     <IfVersion < 2.4>
       Order allow,deny
       Allow from all
     </IfVersion>
   </Directory>
</VirtualHost>
```

Enable the Identity service virtual hosts:

```
ln -s /etc/apache2/sites-available/wsgi-keystone.conf /etc/apache2/sites-enabled
rm -f /var/lib/keystone/keystone.db
```

Start the server:

```
        service apache2 start
```

Export Authentication Information:

```
        export OS_TOKEN=9cad17371f842705310e
        export OS_URL=http://controller:35357/v3
        export OS_IDENTITY_API_VERSION=3
```

Create the service entity for the Identity service

```
        openstack service create --name keystone --description "OpenStack Identity" identity
```

Create the Identity service API endpoints:

```
openstack endpoint create --region RegionOne identity public http://controller:5000/v2.0
openstack endpoint create --region RegionOne identity internal http://controller:5000/v2.0
openstack endpoint create --region RegionOne identity admin http://controller:35357/v2.0
```

Create projects, users, and roles (the password should be the same as previous configuration):

Create the admin project:

```
openstack project create --domain default --description "Admin Project" admin
```

Create the admin user:

```
openstack user create --domain default --password-prompt admin
```

Create the admin role:

```
openstack role create admin
```

Add the admin role to the admin project and user:

```
openstack role add --project admin --user admin admin
```

Create the `service` and demo projects (Tenants):

```
openstack project create --domain default --description "Service Project" service
```

```
openstack project create --domain default --description "Demo Project" demo
```

Create the `demo` user:

```
openstack user create --domain default --password-prompt demo
```

Create the user role:

```
openstack role create user
```

Add the user role to the demo project and user

```
openstack role add --project demo --user demo user
```

Edit /etc/keystone/keystone-paste.ini

Remove ***admin_token_auth*** from these sections **[pipeline:public_api], [pipeline:admin_api],** and **[pipeline:api_v3]**

As the `admin` user, request an authentication token:

```
openstack --os-auth-url http://controller:35357/v3 --os-project-domain-id
default --os-user-domain-id default --os-project-name admin --os-username
admin --os-auth-type password token issue
```

As the `demo` user, request an authentication token:

```
openstack --os-auth-url http://controller:5000/v3 --os-project-domain-id default
--os-user-domain-id default --os-project-name demo --os-username demo
--os-auth-type password token issue
```

Create these two files for authentication:

Edit in admin-openrc.sh:

```
Export OS_PROJECT_DOMAIN_ID=default
Export OS_USER_DOMAIN_ID=default
Export OS_PROJECT_NAME=admin
Export OS_TENANT_NAME=admin
Export OS_USERNAME=admin
Export OS_PASSWORD=P@ssw0rd
Export OS_AUTH_TYPE=password
Export OS_AUTH_URL=http://controller:35357/v3
Export OS_IDENTITY_API_VERSION=3
```

Edit the demo-openrc.sh:

```
Export OS_PROJECT_DOMAIN_ID=default
Export OS_USER_DOMAIN_ID=default
Export OS_PROJECT_NAME=demo
Export OS_TENANT_NAME=demo
Export OS_USERNAME=demo
Export OS_PASSWORD=P@ssw0rd
Export OS_AUTH_TYPE=password
Export OS_AUTH_URL=http://controller:5000/v3
```

```
        Export OS_IDENTITY_API_VERSION=3
```

Source the admin file and issue a token:

```
        source admin-openrc.sh
        openstack token issue
```

Glance Configuration:

```
        mysql -u root -p
        CREATE DATABASE glance;
        GRANT ALL PRIVILEGES ON glance.* TO 'glance'@'localhost' IDENTIFIED BY 'P@ssw0rd';
        GRANT ALL PRIVILEGES ON glance.* TO 'glance'@'%' IDENTIFIED BY 'P@ssw0rd';
```

Quick check:

```
        mysql -h localhost -P 1377 -D glance -u glance -pP@ssw0rd
```

Create a user for glance:

```
        source admin-openrc.sh
        openstack user create --domain default --password-prompt glance
```

Add the admin role to the glance user and service project:

```
        openstack role add --project service --user glance admin
```

Create the glance service entity:

```
        openstack service create --name glance --description "OpenStack Image service" image
```

Create the Image service API endpoints:

```
openstack endpoint create --region RegionOne image public http://controller:9292

openstack endpoint create --region RegionOne image internal http://controller:9292

openstack endpoint create --region RegionOne image admin http://controller:9292
```

Install glance packages:

```
        apt-get install glance python-glanceclient
```

Edit /etc/glance/glance-api.conf:

```
[DEFAULT]
notification_driver = noop
[database]
connection = mysql+pymysql://glance:P@ssw0rd@controller/glance
[keystone_authtoken]
auth_uri = http://controller:5000
auth_url = http://controller:35357
auth_plugin = password
project_domain_id = default
user_domain_id = default
project_name = service
username = glance
password = P@ssw0rd
[paste_deploy]
flavor = keystone
[glance_store]
default_store = file
filesystem_store_datadir = /var/lib/glance/images/
```

Comment out these lines

```
#sqlite_db = /var/lib/glance/glance.sqlite
#backend = sqlalchemy
```

Edit /etc/glance/glance-registry.conf:

```
[DEFAULT]
notification_driver = noop
[database]
connection = mysql+pymysql://glance:P@ssw0rd@controller/glance
[keystone_authtoken]
auth_uri = http://controller:5000
auth_url = http://controller:35357
auth_plugin = password
```

```
project_domain_id = default
user_domain_id = default
project_name = service
username = glance
password = P@ssw0rd
[paste_deploy]
flavor = keystone
```

Comment out these lines

```
#sqlite_db = /var/lib/glance/glance.sqlite
#backend = sqlalchemy
```

Populate the Image service database

```
/bin/sh -c "glance-manage db_sync" glance
rm -f /var/lib/glance/glance.sqlite
```

Restart the service:

```
service glance-registry restart && service glance-api restart
```

Configure the Image service client to use API version 2.0

```
echo "export OS_IMAGE_API_VERSION=2" | tee -a admin-openrc.sh demo-openrc.sh
```

Download cirrosOs image and upload it to glance

```
wget http://download.cirros-cloud.net/0.3.4/cirros-0.3.4-x86_64-disk.img

glance image-create --name "cirros" --file cirros-0.3.4-x86_64-disk.img --disk-format qcow2 --container-format bare --visibility public --progress
```

Confirm the upload:

```
    glance image-list
```

Nova Configuration:

Create Nova Database:

```
    mysql -u root -p
    CREATE DATABASE nova;
    GRANT ALL PRIVILEGES ON nova.* TO 'nova'@'localhost' IDENTIFIED BY 'P@ssw0rd';
    GRANT ALL PRIVILEGES ON nova.* TO 'nova'@'%' IDENTIFIED BY 'P@ssw0rd';
```

Create the user and service entity and API endpoints:

```
    source admin-openrc.sh
    openstack user create --domain default --password-prompt nova
```

Add the admin role to the nova user and service project:

```
    openstack role add --project service --user nova admin
```

Create the nova service entity:

```
    openstack service create --name nova --description "OpenStack Compute" compute
```

Create the Compute service API endpoints:

```
    openstack endpoint create --region RegionOne compute public http://controller:8774/v2/%\(tenant_id\)s
    openstack endpoint create --region RegionOne compute internal http://controller:8774/v2/%\(tenant_id\)s
    openstack endpoint create --regionRegionOne compute admin http://controller:8774/v2/%\(tenant_id\)s
```

Installing the proper packages:

```
    apt-get install nova-api nova-cert nova-conductor nova-consoleauth nova-novncproxy nova-scheduler python-novaclient
```

Edit the /etc/nova/nova.conf file:

```
[DEFAULT]
rpc_backend = rabbit
auth_strategy = keystone
my_ip = 192.168.151.9
network_api_class = nova.network.neutronv2.api.API
security_group_api = neutron
linuxnet_interface_driver = nova.network.linux_net.NeutronLinuxBridgeInterfaceDriver
```

```
firewall_driver = nova.virt.firewall.NoopFirewallDriver
enabled_apis=osapi_compute,metadata
[database]
connection = mysql+pymysql://nova:P@ssw0rd@controller/nova
[oslo_messaging_rabbit]
rabbit_host = controller
rabbit_userid = openstack
rabbit_password = P@ssw0rd
[keystone_authtoken]
auth_uri = http://controller:5000
auth_url = http://controller:35357
auth_plugin = password
project_domain_id = default
user_domain_id = default
project_name = service
username = nova
password = P@ssw0rd
[vnc]
vncserver_listen = $my_ip
vncserver_proxyclient_address = $my_ip
novncproxy_base_url = http://controller:6080/vnc_auto.html
enabled = True
[glance]
host = controller
[oslo_concurrency]
lock_path = /var/lib/nova/tmp
```

Populate the Nova database:

```
sh -c "nova-manage db sync" nova
```

Restart the Nova services:

```
        rm -f /var/lib/nova/nova.sqlite
        service nova-api restart
        service nova-cert restart
        service nova-consoleauth restart
        service nova-scheduler restart
        service nova-conductor restart
        service nova-novncproxy restart
```

Verify nova operations:

```
source admin-openrc.sh
nova service-list
nova image-list
```

Install nova-compute package:

```
apt-get install nova-compute sysfsutils
```

Edit the [libvirt] section in the /etc/nova/nova-compute.conf file:

```
[libvirt]
virt_type = qemu
```

Restart the Compute service:

```
rm -f /var/lib/nova/nova.sqlite
service nova-compute restart
```

Verify the operations:

```
source admin-openrc.sh
nova service-list
```

Neutron Configuration:

Create Neutron Database:

```
mysql -u root -p
CREATE DATABASE neutron;
GRANT ALL PRIVILEGES ON neutron.* TO 'neutron'@'localhost' IDENTIFIED BY 'P@ssw0rd';
GRANT ALL PRIVILEGES ON neutron.* TO 'neutron'@'%' IDENTIFIED BY 'P@ssw0rd';
```

Create the user and service entity and API endpoints:

```
openstack user create --domain default --password-prompt neutron
openstack role add --project service --user neutron admin
```

Create the neutron service entity:

```
openstack service create --name neutron --description "OpenStack Networking" network
```

Create the Compute service API endpoints:

```
        openstack endpoint create --region RegionOne network public
http://controller:9696
        openstack endpoint create --region RegionOne network internal
http://controller:9696
        openstack endpoint create --region RegionOne network admin
http://controller:9696
```

Install packages:

```
        apt-get install neutron-server neutron-plugin-ml2
neutron-plugin-linuxbridge-agent neutron-l3-agent neutron-dhcp-agent
neutron-metadata-agent python-neutronclient
```

Edit the /etc/neutron/neutron.conf file:

```
[DEFAULT]
core_plugin = ml2
service_plugins = router
allow_overlapping_ips = True
rpc_backend = rabbit
auth_strategy = keystone
notify_nova_on_port_status_changes = True
notify_nova_on_port_data_changes = True
nova_url = http://controller:8774/v2

[database]
connection = mysql+pymysql://neutron:P@ssw0rd@controller/neutron

[oslo_messaging_rabbit]
rabbit_host = controller
rabbit_userid = openstack
rabbit_password = P@ssw0rd
[keystone_authtoken]
auth_uri = http://controller:5000
auth_url = http://controller:35357
auth_plugin = password
project_domain_id = default
user_domain_id = default
project_name = service
username = neutron
```

```
password = P@ssw0rd
```

#Comment out or remove any other options in the [keystone_authtoken] section.

```
[nova]
auth_url = http://controller:35357
auth_plugin = password
project_domain_id = default
user_domain_id = default
region_name = RegionOne
project_name = service
username = nova
password = P@ssw0rd
```

Edit the /etc/neutron/plugins/ml2/ml2_conf.ini file

```
[ml2]
type_drivers = flat,vlan,vxlan
tenant_network_types = vxlan
mechanism_drivers = linuxbridge,l2population
extension_drivers = port_security

[ml2_type_flat]
flat_networks = public

[securitygroup]
enable_ipset = True
```

Configure the Linux bridge agent on the Network node
Run ifconfig and find the name of the current active interface that is connected to the interface (in this example, it's em1).

Edit the /etc/neutron/plugins/ml2/linuxbridge_agent.ini file

```
[linux_bridge]
physical_interface_mappings = public:em1

[vxlan]
enable_vxlan = True
local_ip = 192.168.151.9
l2_population = True
```

```
[agent]
prevent_arp_spoofing = True

[securitygroup]
enable_security_group = True
firewall_driver = neutron.agent.linux.iptables_firewall.IptablesFirewallDriver
```

Edit the /etc/neutron/l3_agent.ini file

```
[DEFAULT]
interface_driver = neutron.agent.linux.interface.BridgeInterfaceDriver
external_network_bridge =
verbose = True
```

Configure the DHCP agent

Edit the /etc/neutron/dhcp_agent.ini file

```
[DEFAULT]
interface_driver = neutron.agent.linux.interface.BridgeInterfaceDriver
dhcp_driver = neutron.agent.linux.dhcp.Dnsmasq
enable_isolated_metadata = True
verbose = True
dnsmasq_config_file = /etc/neutron/dnsmasq-neutron.conf
```

Create /etc/neutron/dnsmasq-neutron.conf

```
dhcp-option-force=26,1450
# optionallyadd logging parameters
log-facility = /var/log/neutron/dnsmasq.log
log-dhcp
```

Edit /etc/neutron/metadata_agent.ini

```
[DEFAULT]
auth_uri = http://controller:5000
auth_url = http://controller:35357
auth_region = RegionOne
auth_plugin = password
project_domain_id = default
```

```
user_domain_id = default
project_name = service
username = neutron
password = P@ssw0rd
nova_metadata_ip = controller
metadata_proxy_shared_secret = P@ssw0rd
verbose = True
```

Edit the /etc/nova/nova.conf file

```
[neutron]
url = http://controller:9696
auth_url = http://controller:35357
auth_plugin = password
project_domain_id = default
user_domain_id = default
region_name = RegionOne
project_name = service
username = neutron
password = P@ssw0rd
service_metadata_proxy = True
metadata_proxy_shared_secret = P@ssw0rd
```

Populate the Database for neutron

```
/bin/sh -c "neutron-db-manage --config-file /etc/neutron/neutron.conf --config-file /etc/neutron/plugins/ml2/ml2_conf.ini upgrade head" neutron
```

Restart the services

```
service nova-api restart
service neutron-server restart
service neutron-plugin-linuxbridge-agent restart
service neutron-dhcp-agent restart
service neutron-metadata-agent restart
service neutron-l3-agent restart
rm -f /var/lib/neutron/neutron.sqlite
```

Verify Neutron operation

Run these commands to verify:

```
neutron ext-list
neutron agent-list
```

Create public network:

```
neutron net-create public --shared --provider:physical_network public
--provider:network_type flat
```

```
neutron subnet-create public 10.76.246.0/24 --name public --allocation-pool
start=10.76.246.120,end=10.76.246.139 --dns-nameserver 10.76.246.101 --gateway
10.76.246.1
```

Create private network:

```
source demo-openrc.sh
```

```
neutron net-create private
```

```
neutron subnet-create private 192.168.5.0/24 --name private --dns-nameserver
10.76.246.101 --gateway 192.168.5.1
```

Router creation:

```
neutron net-update public --router:external
```

```
neutron router-create router
```

```
neutron router-interface-add router private
```

```
neutron router-gateway-set router public
```

Verify Network Operations:

```
Neutron net-list
```

```
neutron router-port-list router
```

Instance Creation:

Generate ssh key:

```
ssh-keygen -q -N ""
```

```
nova keypair-add --pub-key .ssh/id_rsa.pub mykey
nova keypair-list
```

Add security group rules:

```
nova secgroup-add-rule default icmp -1 -1 0.0.0.0/0

nova secgroup-add-rule default tcp 22 22 0.0.0.0/0
```

Launch Instance:

```
nova flavor-list
nova image-list
neutron net-list
nova secgroup-list

nova boot --flavor m1.tiny --image cirros --nic
net-id=3ea75491-041d-4587-b439-ad68583c46ad --security-group default --key-name
mykey public-instance

nova list

nova get-vnc-console public-instance novnc
```