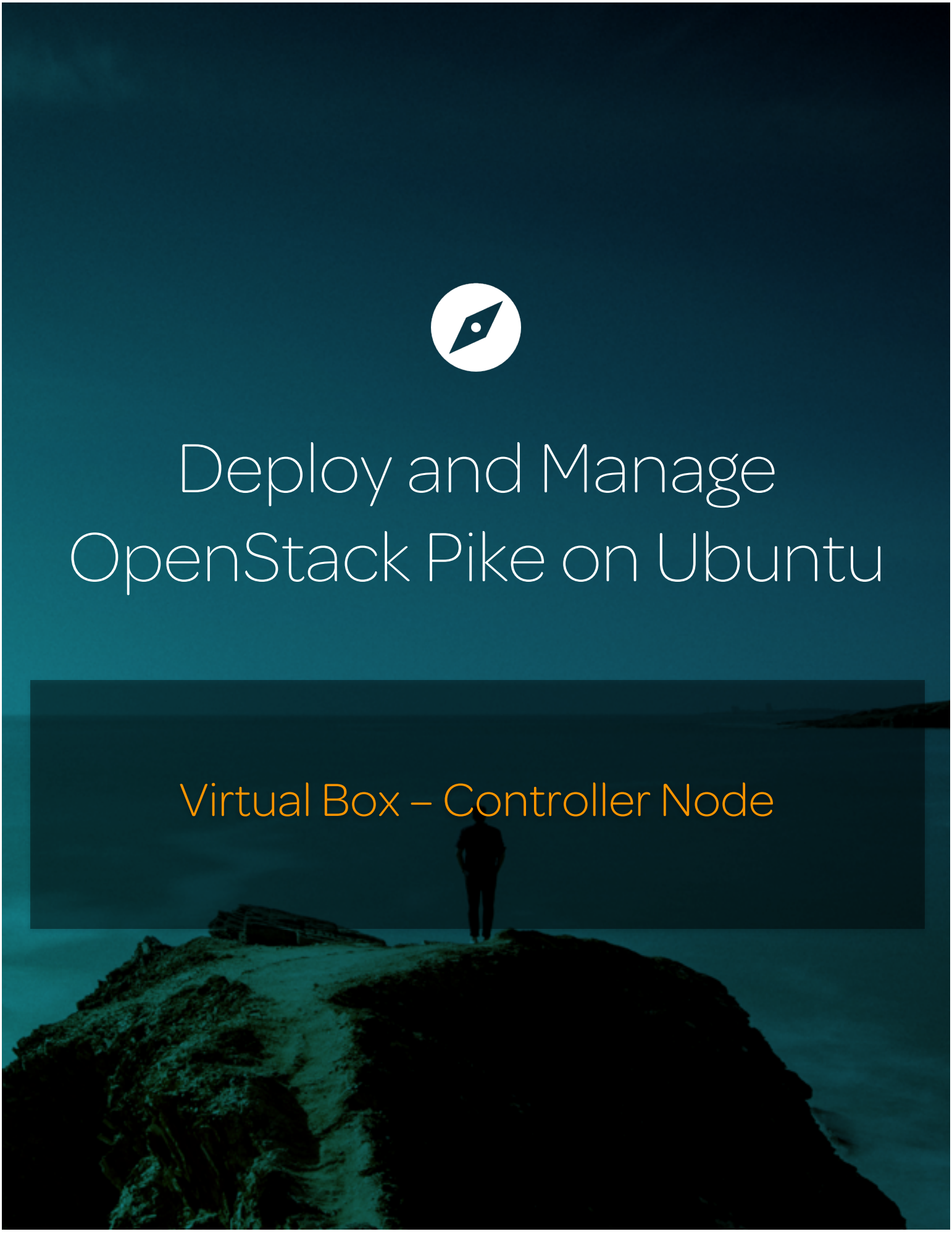


Deploy and Manage OpenStack Pike on Ubuntu

Virtual Box – Controller Node

Creating our Controller

Node Specifications

CPU – 1 processor
RAM – 4 GB
Storage – 10 GB

Controller Node Networking

/etc/network/interfaces

```
auto enp0s8
iface enp0s8 inet static
address 10.0.0.11
netmask 255.255.255.0

auto enp0s9
iface enp0s9 inet static
address 203.0.113.11
netmask 255.255.255.0
```

Controller Node Hosts File

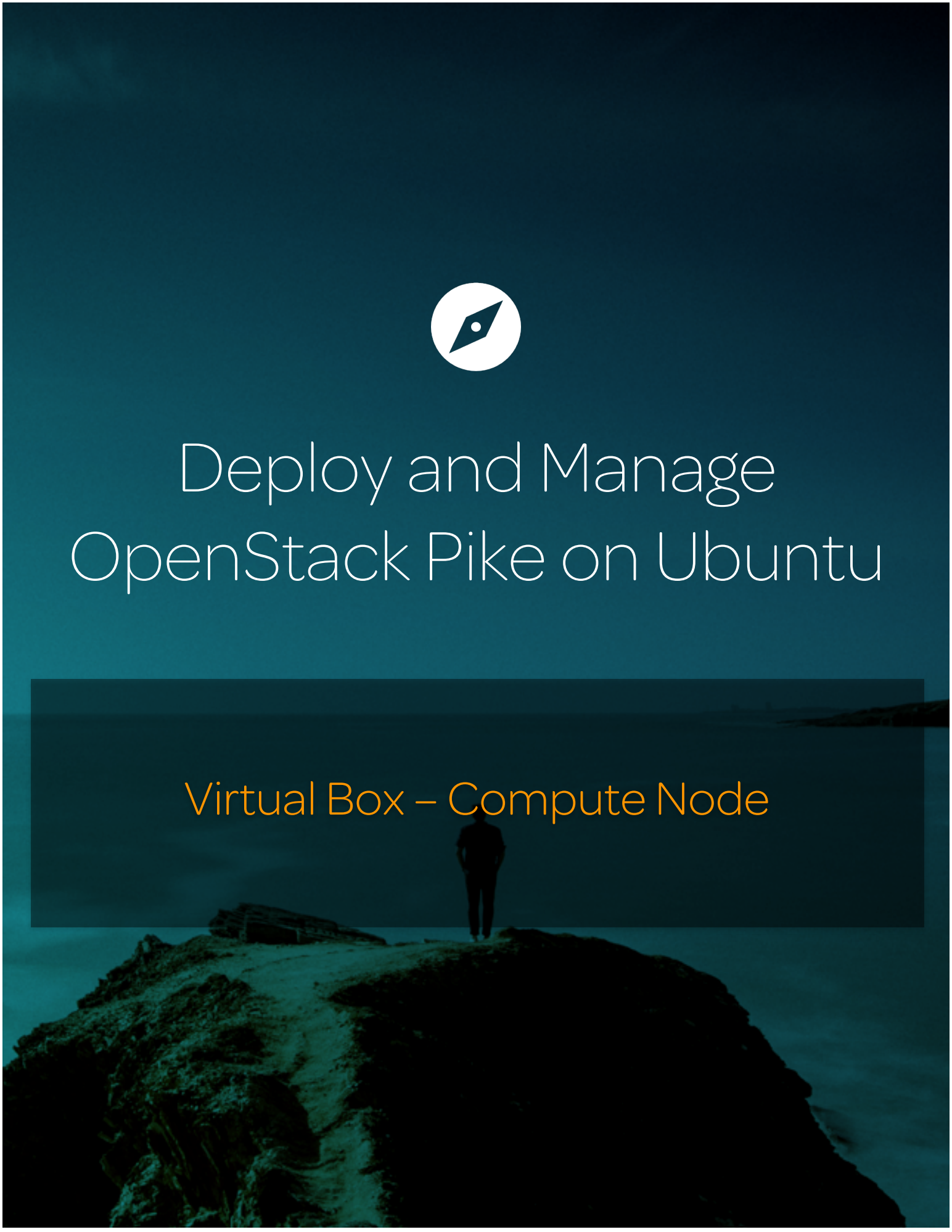
/etc/hosts

10.0.0.11 controller
203.0.113.11 controller-api

10.0.0.31 compute
203.0.113.31 compute-api

10.0.0.41 block1

10.0.0.51 object1
10.0.0.52 object2



Deploy and Manage OpenStack Pike on Ubuntu

Virtual Box – Compute Node

Creating our Compute

Node Specifications

CPU – 1 processor
RAM – 4 GB
Storage – 10 GB

Compute Node Networking

/etc/network/interfaces

```
auto enp0s8
iface enp0s8 inet static
address 10.0.0.31
netmask 255.255.255.0

auto enp0s9
iface enp0s9 inet static
address 203.0.113.31
netmask 255.255.255.0
```

Compute Node Hosts File

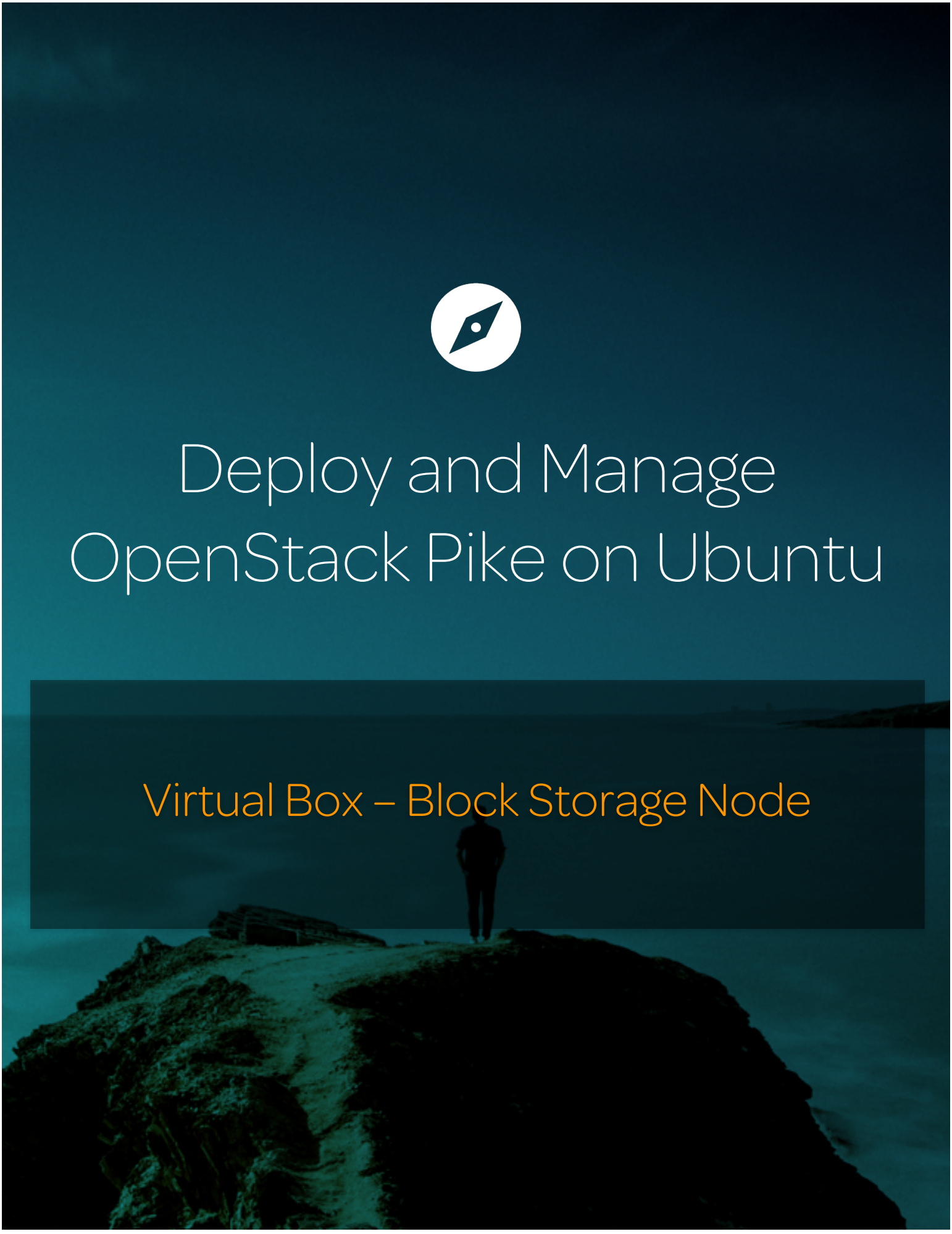
/etc/hosts

10.0.0.11 controller
203.0.113.11 controller-api

10.0.0.31 compute
203.0.113.31 compute-api

10.0.0.41 block1

10.0.0.51 object1
10.0.0.52 object2



Deploy and Manage OpenStack Pike on Ubuntu

Virtual Box – Block Storage Node

Creating our Block Storage Node

Node Specifications

CPU – 1 processor
RAM – 2 GB
Storage – 8 GB drive 1 and 10 GB drive 2

Block Storage Node Networking

/etc/network/interfaces

```
auto enp0s8
iface enp0s8 inet static
address 10.0.0.41
netmask 255.255.255.0
```

Block Storage Node Hosts File

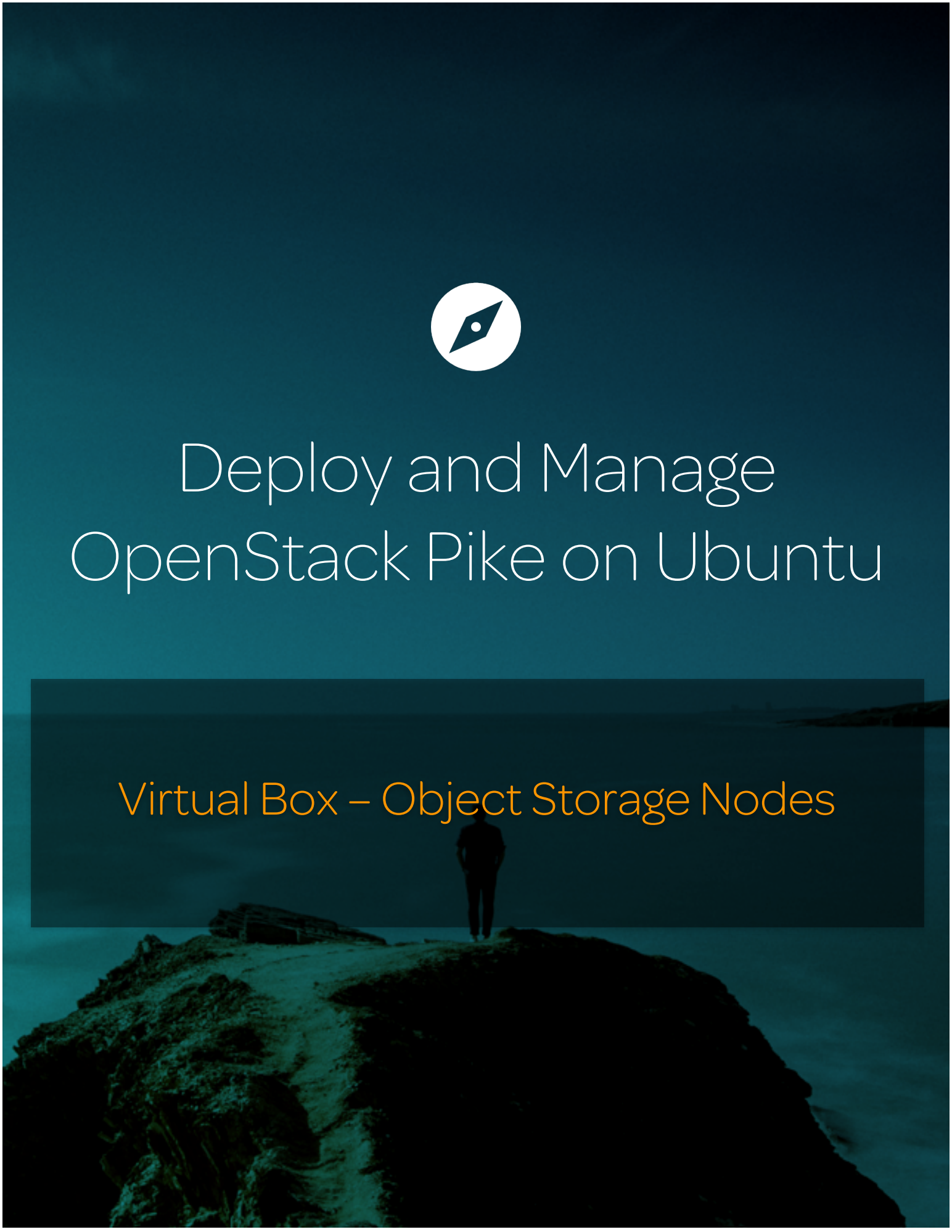
/etc/hosts

10.0.0.11 controller
203.0.113.11 controller-api

10.0.0.31 compute
203.0.113.31 compute-api

10.0.0.41 block1

10.0.0.51 object1
10.0.0.52 object2



Deploy and Manage OpenStack Pike on Ubuntu

Virtual Box – Object Storage Nodes

Creating our Object Storage Nodes

Node Specifications

CPU – 1 processor
RAM – 2 GB
Storage – 8 GB drive 1, 8 GB drive 2,
and 8 GB drive 3

Object Storage Node Networking

/etc/network/interfaces (each node)

On object1
```
auto enp0s8
iface enp0s8 inet static
address 10.0.0.51
netmask 255.255.255.0
```

On object2
```
auto enp0s8
iface enp0s8 inet static
address 10.0.0.52
netmask 255.255.255.0
```

Object Storage Node Hosts File

/etc/hosts

```
10.0.0.11 controller
203.0.113.11 controller-api

10.0.0.31 compute
203.0.113.31 compute-api

10.0.0.41 block1

10.0.0.51 object1
10.0.0.52 object2
```

Deploy and Manage OpenStack Pike on Ubuntu

Prepping the Environment

Passwords

There are two main types of passwords we will be setting in this course:

- SERVICE_PASS
- SERVICE_DBPASS

We will be using two default passwords through out this course:

openstack

- For ADMIN_PASS, aka the admin user password
- For the demo user password

linuxacademy123

- For rabbitmq with the openstack user
- For all SERVICE_PASS
- For all SERVICE_DBPASS

Configuring NTP

Controller Node

```
apt install chrony
nano /etc/chrony/chrony.conf
        allow 10.0.0.0/24
service chrony restart
```

Other Nodes

```
apt install chrony
nano /etc/chrony/chrony.conf
        #pool 2.debian.pool.ntp.org offline iburst
        server controller iburst
service chrony restart
```

Verify All Nodes

```
chronyc sources
```

OpenStack Packages

All Nodes

```
apt install software-properties-common
add-apt-repository cloud-archive:pike
add-apt-repository cloud-archive:pike-updates

apt update && apt dist-upgrade

apt install python-openstackclient
```

Installing the Database

Controller

```
apt install mariadb-server python-pymysql

nano /etc/mysql/mariadb.conf.d/99-openstack.cnf
    [mysqld]
    bind-address = 10.0.0.11
    default-storage-engine = innodb
    innodb_file_per_table = on
    max_connections = 4096
    collation-server = utf8_general_ci
    character-set-server = utf8

service mysql restart
mysql_secure_installation
```

Installing the Message Queue

Controller

```
apt install rabbitmq-server
```

```
rabbitmqctl add_user openstack RABBIT_PASS
```

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

Installing Memcached

Controller

apt install memcached python-memcache

nano /etc/memcached.conf

```
#-l 127.0.0.1
-l 10.0.0.11
```

service memcached restart

Deploy and Manage OpenStack Pike on Ubuntu

Installing Keystone

Installing Keystone

Database

Update python-pyasn1:

```
apt install python-pyasn1
```

Create Keystone database:

```
mysql
create database keystone;
```

Grant access:

```
grant all privileges on keystone.* to 'keystone'@'localhost'
identified by 'linuxacademy123';
grant all privileges on keystone.* to 'keystone'@'%' identified by
'linuxacademy123';

exit
```

Installing Keystone

Install Components

```
apt install keystone apache2 libapache2-mod-wsgi
```

nano /etc/keystone/keystone.conf

```
[database]
# ...
connection =
mysql+pymysql://keystone:linuxacademy123@controller/keystone

[token]
# ...
provider = fernet
```

Populate the database

```
su -s /bin/sh -c “keystone-manage db_sync” keystone
```

Installing Keystone

Initialize Fernet key repositories

```
keystone-manage fernet_setup --keystone-user keystone \
--keystone-group keystone
keystone-manage credential_setup --keystone-user keystone \
--keystone-group keystone
```

Bootstrap Identity Service

```
keystone-manage bootstrap --bootstrap-password openstack \
--bootstrap-admin-url http://controller:35357/v3/ \
--bootstrap-internal-url http://controller:5000/v3/ \
--bootstrap-public-url http://controller:5000v3/ \
--bootstrap-region-id RegionOne
```

Configure Apache

nano /etc/apache2/apache2.conf:

```
ServerName controller
```

Restart Apache:

```
service apache2 restart
```

Creating Projects, Users and Roles

Set Credentials

```
export OS_USERNAME=admin
export OS_PASSWORD=openstack
export OS_PROJECT_NAME=admin
export OS_USER_DOMAIN_NAME=Default
export OS_PROJECT_DOMAIN_NAME=Default
export OS_AUTH_URL=http://controller:35357/v3
export OS_IDENTITY_API_VERSION=3
```

Create Projects

Service Project:

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

Demo Project:

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

Creating Projects, Users and Roles

Create the demo user

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

Create the user role

openstack role create user

Add the user role

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

Verify Installation

Disable Temporary Auth

nano /etc/keystone/keystone-paste.ini:
Remove admin_token_auth from:
[pipeline:public_api]
[pipeline:admin_api]
[pipeline:api_v3]

Unset Environment Variables

unset OS_AUTH_URL OS_PASSWORD

Verify Installation

Request Token

Admin:

```
openstack --os-auth-url http://controller:35357/v3 \
--os-project-domain-name Default --os-user-domain-name \
Default --os-project-name admin --os-username admin \
token issue
```

Demo:

```
openstack --os-auth-url http://controller:5000/v3 \
--os-project-domain-name Default --os-user-domain-name \
Default --os-project-name demo --os-username demo \
token issue
```

Create Credential Files

nano admin-openrc

```
export OS_USERNAME=admin
export OS_PASSWORD=openstack
export OS_PROJECT_NAME=admin
export OS_USER_DOMAIN_NAME=Default
export OS_PROJECT_DOMAIN_NAME=Default
export OS_AUTH_URL=http://controller:35357/v3
export OS_IDENTITY_API_VERSION=3
export OS_IMAGE_API_VERSION=2
```

nano demo-openrc

```
export OS_USERNAME=demo
export OS_PASSWORD=openstack
export OS_PROJECT_NAME=demo
export OS_USER_DOMAIN_NAME=Default
export OS_PROJECT_DOMAIN_NAME=Default
export OS_AUTH_URL=http://controller:5000/v3
export OS_IDENTITY_API_VERSION=3
export OS_IMAGE_API_VERSION=2
```

Using the Credentials Files

Using admin-openrc

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

Using demo-openrc

```
. demo-openrc
openstack token issue
```

Deploy and Manage OpenStack Pike on Ubuntu

Installing Glance

Installing Glance

Database

Create Glance Database:

```
mysql
create database glance;
```

Grant Access:

```
grant all privileges on glance.* to 'glance'@'localhost' identified by
'linuxacademy123';
grant all privileges on glance.* to 'glance'@'%' identified by
'linuxacademy123';

exit
```

Setting up the Glance user

Source Admin Credentials

. admin-openrc

Create the Glance User

openstack user create --domain default --password-prompt glance

Add the Admin Role

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

Setting up the Glance Service and Endpoints

Create the Service

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

Create the Endpoints

Public:

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

Internal:

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

Admin:

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

Installing Glance

Install Components

```
apt install glance
```

nano /etc/glance/glance-api.conf

```
[database]
# ...
connection =
mysql+pymysql://glance:linuxacademy123@controller/glance

[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 = glance
password = linuxacademy123
```

Installing Glance

nano /etc/glance/glance-api.conf

```
[paste_deploy]
# ...
flavor = keystone

[glance_store]
# ...
stores = file,http
default_store = file
filesystem_store_directory = /var/lib/glance/images
```

Installing Glance

nano /etc/glance/glance-registry.conf

```
[database]
# ...
connection =
mysql+pymsql://glance:linuxacademy123@controller/glance

[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 = glance
password = linuxacademy123

[paste_deploy]
# ...
flavor = keystone
```

Installing Glance

Populate the Database

su –s /bin/sh –c “glance-manage db_sync” glance

Finalize Installation

service glance-registry restart
service glance-api restart

Verify Installation

Source Credentials

. admin-openrc

Download Image

wget http://download.cirros-cloud.net/0.3.5/cirros-0.3.5-x86_64-disk.img

Upload Image

openstack image create --file cirros-0.3.5-x86_64-disk.img --disk-format qcow2 –container-format bare --public cirros

List Image

openstack image list

Deploy and Manage OpenStack Pike on Ubuntu

Installing Nova

Installing Nova

Database

Create Nova databases:

```
mysql
create database nova_api;
create database nova;
create database nova_cell0;
```

Grant access:

```
grant all privileges on nova_api.* to 'nova'@'localhost'  identified
by 'linuxacademy123';
grant all privileges on nova_api.* to 'nova'@'%' \identified by
'linuxacademy123'
;grant all privileges on nova.* to 'nova'@'localhost' \identified by
'linuxacademy123';
grant all privileges on nova.* to 'nova'@'%' \identified by
'linuxacademy123'`;
grant all privileges on nova_cell0.* to 'nova'@'localhost'
\identified by 'linuxacademy123';
grant all privileges on nova_cell0.* to 'nova'@'%' \identified by
'linuxacademy123'

exit
```

Setting up the Nova User

Source Admin Credentials

. admin-openrc

Create the Nova User

openstack user create --domain default --password-prompt nova

Add the Admin Role

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

Setting up the Nova Service and Endpoints

Create the Service

```
openstack service create --name nova --description
“OpenStack Compute” compute
```

Create the Endpoints

Public:

```
openstack endpoint create --region RegionOne compute \
public http://controller:8774/v2.1
```

Internal:

```
openstack endpoint create --region RegionOne compute \
internal  http://controller:8774/v2.1
```

Admin:

```
openstack endpoint create --region RegionOne compute \
admin http://controller:8774/v2.1
```

Setting up the Placement User

Create the Placement User

openstack user create --domain default --password-prompt placement

Add the Admin Role

openstack role add --project service --user placement admin

Setting up the Placement Service and Endpoints

Create the Service

```
openstack service create --name placement --description
“Placement API” placement
```

Create the Endpoints

Public:

```
openstack endpoint create --region RegionOne \
placement  public http://controller:8778
```

Internal:

```
openstack endpoint create --region RegionOne \
placement internal  http://controller:8778
```

Admin:

```
openstack endpoint create --region RegionOne \
placement admin http://controller:8778
```

Installing Nova

Install Components

apt install nova-api nova-conductor nova-consoleauth nova-novncproxy nova-scheduler nova-placement-api

nano /etc/nova/nova.conf

```
[api_database]
connection =
mysql+pymysql://nova:linuxacademy123@controller/nova_api

[database]
connection =
mysql+pymysql://nova:inuxacademy123@controller/nova

[DEFAULT]
transport_url = rabbit://openstack:linuxacademy123@controller

[api]
auth_strategy = keystone
```

Installing Nova

nano /etc/nova/nova.conf

```
[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 = linuxacademy123

[DEFAULT]
my_ip = 10.0.0.11

[DEFAULT]
#log_dir

[DEFAULT]
use_neutron = True
firewall_driver = nova.virt.firewall.NoopFirewallDrive
```

Installing Nova

nano /etc/nova/nova.conf

```
[vnc]
enabled = true
vncserver_listen = $my_ip
vncserver_proxyclient_address = $my_ip

[glance]
api_servers = http://controller:9292

[oslo_concurrency]
lock_path = /var/lib/nova/tmp

[placement]
auth_url = http://controller:35357/v3
os_region_name=RegionOne
auth_type = password
project_domain_name = default
user_domain_name = default
project_name = service
username = placement
password = linuxacademy123
```

Installing Nova

Populate the nova-api database

su –s /bin/sh –c “nova-manage api_db sync” nova

Register the cell0 databases

su -s /bin/sh -c "nova-manage cell_v2 map_cell0" nova

Create the cell1 cell

su -s /bin/sh -c "nova-manage cell_v2 create_cell --name=cell1 --verbose" nova

Populate the Nova Database

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

Finalize Installation

service nova-api restart
service nova-consoleauth restart
service nova-scheduler restart
service nova-conductor restart
service nova-novncproxy restart

Installing a compute node

Install Components

apt install nova-compute

nano /etc/nova/nova.conf

```
[DEFAULT]
transport_url = rabbit://openstack:linuxacademy123@controller

[api]
auth_strategy = keystone
```

Installing a compute node

nano /etc/nova/nova.conf

```
[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 = linuxacademy123

[DEFAULT]
my_ip = 10.0.0.31

[DEFAULT]
#log_dir

[DEFAULT]
use_neutron = True
firewall_driver = nova.virt.firewall.NoopFirewallDrive
```

Installing a compute node

nano /etc/nova/nova.conf

```
[vnc]
enabled = true
vncserver_listen = 0.0.0.0
vncserver_proxyclient_address = $my_ip

[glance]
api_servers = http://controller:9292

[oslo_concurrency]
lock_path = /var/lib/nova/tmp

[placement]
auth_url = http://controller:35357/v3
os_region_name=RegionOne
auth_type = password
project_domain_name = default
user_domain_name = default
project_name = service
username = placement
password = linuxacademy123
```

Finalize Installation - Compute

Does Compute Node Support Acceleration

```
egrep -c '(vmx|svm)' /proc/cpuinfo
```

nano /etc/nova/nova-compute.conf

```
[libvirt]
# ...
virt_type = qemu
```

Restart Compute Service

```
service nova-compute restart
```

Add Compute to Cell Database

Source Admin Credentials

```
. admin-openrc
```

List Compute Services

```
openstack compute service list
```

Discover Compute Hosts

```
su -s /bin/sh -c "nova-manage cell_v2 discover_hosts --verbose" nova
```

Populate the Nova Database

```
su -s /bin/sh -c "nova-manage db sync" nova
```

Verifying Installation

List API Endpoints

openstack catalog list

Deploy and Manage OpenStack Pike on Ubuntu

Installing Neutron

Installing Neutron

Database

Create Neutron Database:

mysql
create database neutron;

Grant Access:

grant all privileges on neutron.* to 'neutron'@'localhost' identified by 'linuxacademy123';
grant all privileges on neutron.* to 'neutron'@'%' identified by 'linuxacademy123'

exit

Setting up the Neutron User

Source Admin Credentials

. admin-openrc

Create the Neutron User

openstack user create --domain default --password-prompt neutron

Add the Admin Role

openstack role add --project service --user neutron admin

Setting up the Neutron Service and Endpoints

Create the Service

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

Create the Endpoints

Public:

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

Internal:

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

Admin:

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

Installing Neutron

Install Components

apt install neutron-server neutron-plugin-ml2 neutron-linuxbridge-agent neutron-dhcp-agent neutron-metadata-agent

nano /etc/neutron/neutron.conf

```
[database]
connection =
mysql+pymysql://neutron:linuxacademy123@controller/neutron

[DEFAULT]
# ...
core_plugin = ml2
service_plugins =

[DEFAULT]
transport_url = rabbit://openstack:linuxacademy123@controller

[DEFAULT]
auth_strategy = keystone
```

Installing Neutron

nano /etc/neutron/neutron.conf

```
[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 = linuxacademy123

[DEFAULT]
# ...
notify_nova_on_port_status_changes = true
notify_nova_on_port_data_changes = true
```

Installing Neutron

nano /etc/neutron/neutron.conf

```
[nova]
auth_url = http://controller:35357
auth_type = password
project_domain_name = default
user_domain_name = default
project_name = service
username = nova
password = linuxacademy123
```

Installing Neutron

nano /etc/neutron/plugins/ml2/ml2_conf.ini

```
[ml2]
type_drivers = flat,vlan

[ml2]
tenant_network_types =

[ml2]
mechanism_drivers = linuxbridge

[ml2]
extension_drivers = port_security

[ml2_type_flat]
flat_networks = provider

[securitygroup]
enable_ipset = true
```

Installing Neutron

nano /etc/neutron/plugins/ml2/linuxbridge_agent.ini

```
[linux_bridge]
physical_interface_mappings = provider:enp0s9


[vxlan]
enable_vxlan = false

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

Installing Neutron

nano /etc/neutron/dhcp_agent.ini

```
[DEFAULT]
# ...
interface_driver = linuxbridge
dhcp_driver = neutron.agent.linux.dhcp.Dnsmasq
enable_isolated_metadata = true
```

nano /etc/neutron/metadata_agent.ini

```
[DEFAULT]
# ...
nova_metadata_host = 10.0.0.11
metadata_proxy_shared_secret = METADATA_SECRET
```

Installing Neutron

nano /etc/nova/nova.conf

```
[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 = linuxacademy123
service_metadata_proxy = true
metadata_proxy_shared_secret = METADATA_SECRET
```

Installing Neutron

Populate the Neutron Database

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

Finalize Installation

```
service nova-api restart
service neutron-server restart
service neutron-linuxbridge-agent restart
service neutron-dhcp-agent restart
service neutron-metadata-agent restart
```

Installing a Compute Node

Install Components

apt install neutron-linuxbridge-agent

nano /etc/neutron/neutron.conf

```
[DEFAULT]
transport_url = rabbit://openstack:linuxacademy123@controller

[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 = neutron
password = linuxacademy123
```

Installing a Compute Node

nano /etc/neutron/plugins/ml2/linuxbridge_agent.ini

```
[linux_bridge]
physical_interface_mappings = provider:enp0s9


[vxlan]
enable_vxlan = false

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

Installing a Compute Node

nano /etc/nova/nova.conf

```
[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 = linuxacademy123
```

Finalize Installation

```
service nova-compute restart
service neutron-linuxbridge-agent restart
```

Verifying Installation

Source Credentials

```
. admin-openrc
```

List Network Agents

```
openstack network agent list
```

Deploy and Manage OpenStack Pike on Ubuntu

Installing Horizon

Installing Horizon

Install Components

apt install openstack-dashboard

nano /etc/openstack-dashboard/local_settings.py

```
OPENSTACK_HOST = "controller"
OPENSTACK_KEYSTONE_URL = "http://%s:5000/v3" %
OPENSTACK_HOST

OPENSTACK_KEYSTONE_DEFAULT_ROLE = "user"

OPENSTACK_KEYSTONE_DEFAULT_DOMAIN = "Default"


ALLOWED_HOSTS = ['*', ]
```

Installing Horizon

nano /etc/openstack-dashboard/local_settings.py

```
SESSION_ENGINE = 'django.contrib.sessions.backends.cache'
CACHES = {
        'default': {
                'BACKEND':
'django.core.cache.backends.memcached.MemcachedCache',
                'LOCATION': '10.0.0.11:11211',   }}


OPENSTACK_API_VERSIONS = {
        "identity": 3,
        "image": 2,
        "volume": 2,
}

OPENSTACK_KEYSTONE_MULTIDOMAIN_SUPPORT = True
```

Installing Horizon

nano /etc/openstack-dashboard/local_settings.py

```
OPENSTACK_NEUTRON_NETWORK = {
    ...
    'enable_router': False,
    'enable_quotas': False,
    'enable_ipv6': False,
    'enable_distributed_router': False,
    'enable_ha_router': False,
    'enable_lb': False,
    'enable_firewall': False,
    'enable_vpn': False,
    'enable_fip_topology_check': False,

TIME_ZONE = "TIME_ZONE"
```

nano /etc/apache2/conf-available/openstack-dashboard.conf

```
WSGIApplicationGroup %{GLOBAL}
```

Restart compute service

```
service apache2 reload
```

Deploy and Manage OpenStack Pike on Ubuntu

Installing Cinder

Installing Cinder

Database

Create Cinder database:

```
mysql
create database cinder;
```

Grant access:

```
grant all privileges on cinder.* to cinder'@'localhost'  identified by
'linuxacademy123';
grant all privileges on cinder.* to cinder'@'%' identified by
'linuxacademy123'

exit
```

Setting up the Cinder user

Source Admin Credentials

. admin-openrc

Create the Cinder User

openstack user create --domain default --password-prompt cinder

Add the Admin Role

openstack role add --project service --user cinder admin

Setting up the Cinder Services and Endpoints

Create the Services

```
openstack service create --name cinderv2 --description
"OpenStack Block Storage" volumev2
openstack service create --name cinderv3 --description
"OpenStack Block Storage" volumev3
```

Create the Endpoints

Public:

```
openstack endpoint create --region RegionOne volumev2 \
 public  http://controller:8776/v2/%\(project_id\)s
```

Internal:

```
openstack endpoint create --region RegionOne volumev2 \
internal  http://controller:8776/v2/%\(project_id\)s
```

Admin:

```
openstack endpoint create --region RegionOne volumev2 \
admin http://controller:8776/v2/%\(project_id\)s
```

Setting up the Cinder Services and Endpoints

Create the Endpoints

Public:

```
openstack endpoint create --region RegionOne volumev3 \
 public  http://controller:8776/v3/%\(project_id\)s
```

Internal:

```
openstack endpoint create --region RegionOne volumev3 \
internal  http://controller:8776/v3/%\(project_id\)s
```

Admin:

```
openstack endpoint create --region RegionOne volumev3 \
admin http://controller:8776/v3/%\(project_id\)s
```

Installing Cinder

Install Components

apt install cinder-api cinder-scheduler

nano /etc/cinder/cinder.conf

[database]
connection =
mysql+pymysql://cinder:linuxacademy123@controller/cinder

[DEFAULT]
transport_url = rabbit://openstack:linuxacademy123@controller

[api]
auth_strategy = keystone

Installing Cinder

nano /etc/cinder/cinder.conf

```
[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 = cinder
password = linuxacademy123

[DEFAULT]
my_ip = 10.0.0.11

[oslo_concurrency]
lock_path = /var/lib/cinder/tmp
```

Installing Cinder

Populate the Cinder Database

su –s /bin/sh –c “cinder-manage db sync” cinder

nano /etc/nova/nova.conf

[cinder]
os_region_name = RegionOne

Finalize Installation

service nova-api restart
service cinder-scheduler restart
service apache2 restart

Installing a Block Storage Node

Install Components

```
apt install lvm2 thin-provisioning-tools
```

Create the LVM Physical Volume

```
pvcreate /dev/sdb
```

Create the LVM Volume Group

```
vgcreate cinder-volumes /dev/sdb
```

nano /etc/lvm/lvm.conf

```
devices {
 ...
filter = [ "a/sdb/", "r/.*/"]
```

Installing a Block Storage Node

Install Components

apt install cinder-volume

nano /etc/cinder/cinder.conf

[database]
connection =
mysql+pymsql://cinder:linuxacademy123@controller/cinder

[DEFAULT]
transport_url = rabbit://openstack:linuxacademy123@controller

[DEFAULT]
auth_strategy = keystone

Installing a Block Storage Node

nano /etc/cinder/cinder.conf

```
[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 = cinder
password = linuxacademy123

[DEFAULT]
my_ip = 10.0.0.41

[LVM]
volume_driver = cinder.volume.drivers.lvm.LVMVolumeDriver
volume_group = cinder-volumes
iscsi_protocol = iscsi
iscsi_helper = tgtadm
```

Installing a Block Storage Node

nano /etc/cinder/cinder.conf

```
[DEFAULT]
enabled_backends = lvm

[DEFAULT]
glance_api_servers = http://controller:9292

[oslo_concurrency]
lock_path = /var/lib/cinder/tmp
```

Finalize Installation

```
service tgt restart
service cinder-volume restart
```

Verifying Installation

Source Admin Credentials

```
. admin-openrc
```

List Cinder Services

```
openstack volume service list
```

Deploy and Manage OpenStack Pike on Ubuntu

Installing Swift

Installing Swift

Source Admin Credentials

. admin-openrc

Create the Swift User

openstack user create --domain default --password-prompt swift

Add the Admin Role

openstack role add --project service --user swift admin

Setting up the Swift Service and Endpoints

Create the Service

```
openstack service create --name swift --description
“OpenStack Object Storage” object-store
```

Create the Endpoints

Public:

```
openstack endpoint create --region RegionOne object-store \
public http://controller:8080/v1/AUTH_%\(project_id\)s
```

Internal:

```
openstack endpoint create --region RegionOne object-store \
internal http://controller:8080/v1/AUTH_%\(project_id\)s
```

Admin:

```
openstack endpoint create --region RegionOne object-store \
admin http://controller:8080/v1
```

Installing Swift

Install Components

apt install swift swift-proxy python-swiftclient

Create /etc/swift

mkdir /etc/swift

Download the Proxy Service config File

curl -o /etc/swift/proxy-server.conf https://git.openstack.org/cgit/openstack/swift/plain/etc/proxy-server.conf-sample?h=stable/pike

Installing Swift

nano /etc/swift/proxy-server.conf

```
[DEFAULT]
bind_port = 8080
user = swift
swift_dir = /etc/swift

[pipeline:main]
```

Remove tempurl and tempauth modules and add the authtoken and keystoneauth modules

```
[app:proxy-server]
use = egg:swift#proxy
...
account_autocreate = True

[filter:keystoneauth]
use = egg:swift#keystoneauth
 ...
operator_roles = admin,user
```

Installing Swift

nano /etc/swift/proxy-server.conf

```
[filter:authtoken]
paste.filter_factory =
keystonemiddleware.auth_token:filter_factory
...
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 = swift
password = linuxacademy123
delay_auth_decision = True

[filter:cache]
use = egg:swift#memcache
 ...
memcache_servers = controller:11211
```

Installing Swift on the Object Storages Nodes

Install Components

```
apt install xfsprogs rsync
```

Format Devices

```
mkfs.xfs /dev/sdb
mkfs.xfs /dev/sdc
```

Create the Mount Point Directories

```
mkdir -p /srv/node/sdb
mkdir -p /srv/node/sdc
```

nano /etc/fstab

```
/dev/sdb /srv/node/sdb xfs
noatime,nodiratime,nobarrier,logbufs=8 0 2
/dev/sdc /srv/node/sdc xfs
noatime,nodiratime,nobarrier,logbufs=8 0 2
```

Mount the Devices

```
mount /dev/sdb
mount /dev/sdc
```

Installing Swift on the Object Storages Nodes

nano /etc/rsyncd.conf

```
uid = swift
gid = swift
log file = /var/log/rsyncd.log
pid file = /var/run/rsyncd.pid
address = IP OF MACHINE

 [account]
 max connections = 2
path = /srv/node/
read only = False
lock file = /var/lock/account.lock

[container]
max connections = 2
path = /srv/node/
read only = False
lock file = /var/lock/container.lock

[object]
max connections = 2
path = /srv/node/
read only = False
lock file = /var/lock/object.lock
```

Installing Swift on the Object Storages Nodes

nano /etc/default/rsync

RSYNC_ENABLE =true

Restart rsync

service rsync restart

Installing Swift on the Object Storages Nodes

Install Components

apt install swift swift-account swift-container swift-object

Download the Config Files

curl -o /etc/swift/account-server.conf
https://git.openstack.org/cgit/openstack/swift/plain/etc/account-server.conf-sample?h=stable/pike
curl -o /etc/swift/container-server.conf
https://git.openstack.org/cgit/openstack/swift/plain/etc/container-server.conf-sample?h=stable/pike
curl -o /etc/swift/object-server.conf
https://git.openstack.org/cgit/openstack/swift/plain/etc/object-server.conf-sample?h=stable/pike

Installing Swift on the Object Storages nodes

nano /etc/swift/account-server.conf

```
[DEFAULT]
bind_ip = IP OF MACHINE
bind_port = 6202
user = swift
swift_dir = /etc/swift
devices = /srv/node
mount_check = True

[pipeline:main]
pipeline = healthcheck recon account-server

[filter:recon]
use = egg:swift#recon
...
recon_cache_path = /var/cache/swift
```

Installing Swift on the Object Storages Nodes

nano /etc/swift/container-server.conf

```
[DEFAULT]
bind_ip = IP OF MACHINE
bind_port = 6201
user = swift
swift_dir = /etc/swift
devices = /srv/node
mount_check = True

[pipeline:main]
pipeline = healthcheck recon container-server

[filter:recon]
use = egg:swift#recon
...
recon_cache_path = /var/cache/swift
```

Installing Swift on the Object Storages Nodes

nano /etc/swift/object-server.conf

```
[DEFAULT]
bind_ip = IP OF MACHINE
bind_port = 6200
user = swift
swift_dir = /etc/swift
devices = /srv/node
mount_check = True

[pipeline:main]
pipeline = healthcheck recon object-server

[filter:recon]
use = egg:swift#recon
...
recon_cache_path = /var/cache/swift
recon_lock_path = /var/lock
```

Installing Swift on the Object Storages Nodes

Fix Permissions

```
chown swift:swift /srv/node/sdb
chown swift:swift /srv/node/sdc
```

Create Account Ring on Controller

Change Directories

```
cd /etc/swift
```

Create Base account.builder File

```
swift-ring-builder account.builder create 10 3 1
```

Add Each Node and Drive to Ring

```
swift-ring-builder account.builder add --region 1 --zone 1 --ip
10.0.0.51 --port 6202 --device sdb --weight 100
swift-ring-builder account.builder add --region 1 --zone 1 --ip
10.0.0.51 --port 6202 --device sdc --weight 100
swift-ring-builder account.builder add --region 1 --zone 1 --ip
10.0.0.52 --port 6202 --device sdb --weight 100
swift-ring-builder account.builder add --region 1 --zone 1 --ip
10.0.0.52 --port 6202 --device sdc --weight 100
```

Create Account Ring on Controller

Verify the Ring

swift-ring-builder account.builder

Rebalance the Ring

swift-ring-builder account.builder rebalance

Create Container Ring on Controller

Create Base container.builder File

```
swift-ring-builder container.builder create 10 3 1
```

Add Each Node and Drive to Ring

```
swift-ring-builder container.builder add --region 1 --zone 1 --ip
10.0.0.51 --port 6201 --device sdb --weight 100
swift-ring-builder container.builder add --region 1 --zone 1 --ip
10.0.0.51 --port 6201 --device sdc --weight 100
swift-ring-builder container.builder add --region 1 --zone 1 --ip
10.0.0.52 --port 6201 --device sdb --weight 100
swift-ring-builder container.builder add --region 1 --zone 1 --ip
10.0.0.52 --port 6201 --device sdc --weight 100
```

Create Container Ring on Controller

Verify the Ring

swift-ring-builder container.builder

Rebalance the Ring

swift-ring-builder container.builder rebalance

Create Object Ring on Controller

Create Base object.builder File

```
swift-ring-builder object.builder create 10 3 1
```

Add Each Node and Drive to Ring

```
swift-ring-builder object.builder add --region 1 --zone 1 --ip
10.0.0.51 --port 6200 --device sdb --weight 100
swift-ring-builder object.builder add --region 1 --zone 1 --ip
10.0.0.51 --port 6200 --device sdc --weight 100
swift-ring-builder object.builder add --region 1 --zone 1 --ip
10.0.0.52 --port 6200 --device sdb --weight 100
swift-ring-builder object.builder add --region 1 --zone 1 --ip
10.0.0.52 --port 6200 --device sdc –weight1 100
```

Create Object Ring on Controller

Verify the Ring

swift-ring-builder object.builder

Rebalance the Ring

swift-ring-builder object.builder rebalance

Distribute Ring Files

Accept Keys if Needed

ssh 10.0.0.51
ssh 10.0.0.52

Distribute the Ring Files

for x in 10.0.0.51 10.0.0.52 ; do scp *.ring.gz amy@$x:/tmp/;done

Move Ring Files to /etc/swift on Each Node

mv /tmp/*.ring.gz /etc/swift

Finalizing Swift

Obtain swift.conf

curl -o /etc/swift/swift.conf
https://git.openstack.org/cgit/openstack/swift/plain/etc/swift.conf-sample?h=stable/pike

Create 2 Hash Secrets

openssl rand –hex 6

nano /etc/swift/swift.conf

[swift-hash]
...
swift_hash_path_suffix = HASH_PATH_SUFFIX
swift_hash_path_prefix = HASH_PATH_PREFIX

[storage-policy:0]
...
name = Policy-0
default = yes

Finalizing Swift

Distribute swift.conf

```
for x in 10.0.0.51 10.0.0.52 ; do scp /etc/swift/swift.conf
amy@$x:/tmp/;done
```

Move Ring Files to /etc/swift on Each Node

```
mv /tmp/swift.conf /etc/swift
```

Ensure Proper Ownership

```
chown -R root:swift /etc/swift
```

Restart Services on Controller

```
service memcached restart
service swift-proxy restart
```

Start Services on Object Storage Nodes

```
swift-init all start
```

Verifying Installation

Source Demo Credentials

```
. demo-openrc
```

Show Status

```
swift stat
```

Create Container

```
openstack container create container1
```

Create testfile

```
echo 'Mary had a little lamb' > testfile
```

Upload a Test File

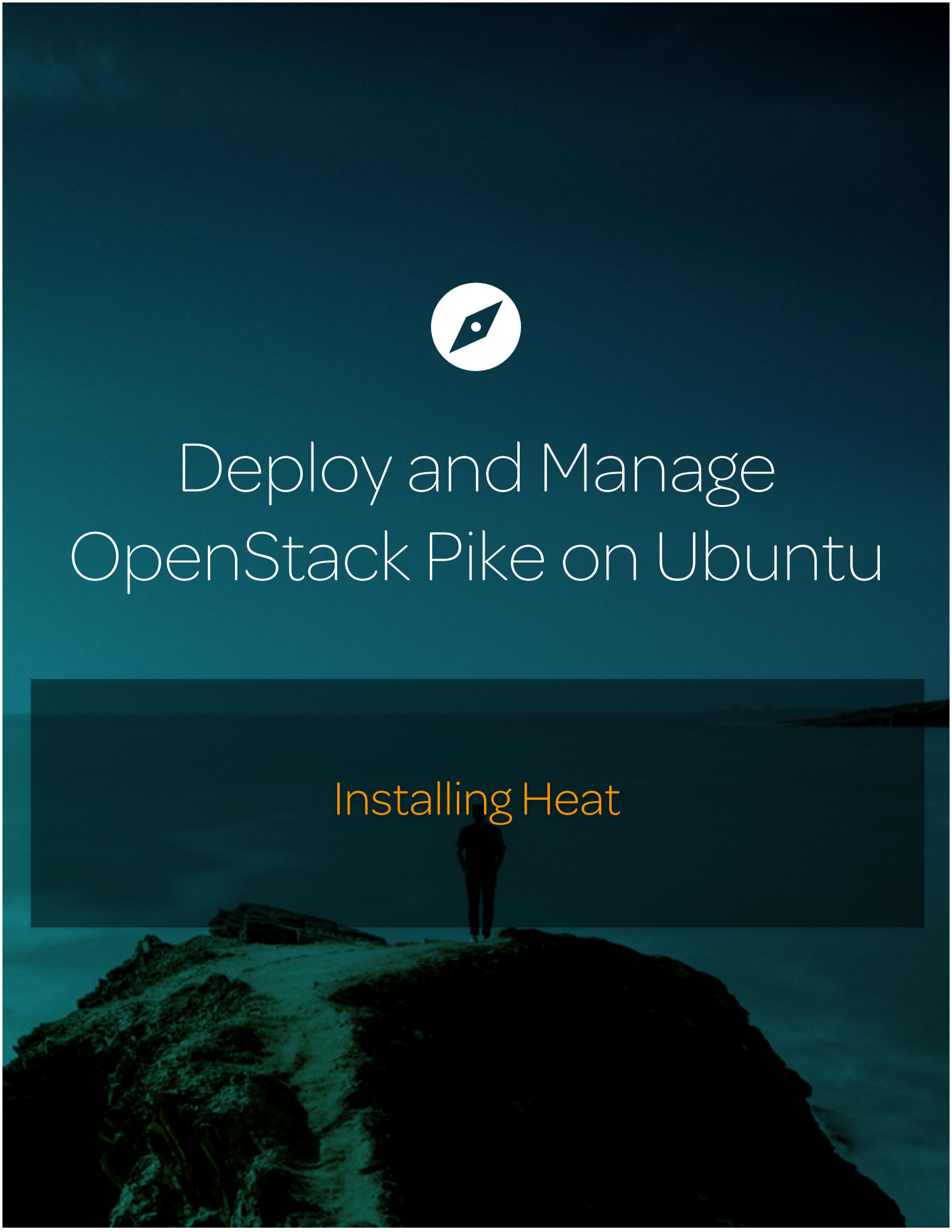
```
openstack object create container1 testfile
```

List Files in Container

```
openstack object list container1
```

Download File from Container

```
openstack object save container1 testfile
```



Deploy and Manage OpenStack Pike on Ubuntu

Installing Heat

Installing Heat

Database

Create heat database:

```
mysql
create database heat;
```

Grant access:

```
grant all privileges on heat.* to 'heat'@'localhost' identified by
'linuxacademy123';
grant all privileges on heat.* to 'heat'@'%' identified by
'linuxacademy123';

exit
```

Setting up the Heat User

Source Admin Credentials

. admin-openrc

Create the Heat User

openstack user create --domain default --password-prompt heat

Add the Admin Role

openstack role add --project service --user heat admin

Setting up the Heat Service and Endpoints

Create the Service

```
openstack service create --name heat --description
"Orchestration" orchestration
openstack service create --name heat-cfn --description
"Orchestration" cloudformation
```

Create the Endpoints

Public:

```
openstack endpoint create --region RegionOne orchestration \
 public http://controller:8004/v1/%\(tenant_id\)s
```

Internal:

```
openstack endpoint create --region RegionOne orchestration \
internal  http://controller:8004/v1/%\(tenant_id\)s
```

Admin:

```
openstack endpoint create --region RegionOne orchestration \
admin http://controller:8004/v1/%\(tenant_id\)s
```

Setting up the Heat Service and Endpoints

Create the Endpoints

Public:

```
openstack endpoint create --region RegionOne \
cloudformation public http://controller:8000/v1
```

Internal:

```
openstack endpoint create --region RegionOne \
cloudformation internal  http://controller:8000/v1
```

Admin:

```
openstack endpoint create --region RegionOne \
cloudformation admin http://controller:8000/v1
```

Additional Identity Information

Create the Heat Domain

openstack domain create --description “Stack projects and users” heat

Create the heat_domain_user user

openstack user create --domain heat --password-prompt heat_domain_admin

Add the Admin Role

openstack role add --domain heat --user-domain heat --user heat_domain_admin admin

Create the heat_stack_owner role

openstack role create heat_stack_owner

Add the heat_stack_owner role

openstack role add --domain heat --user demo heat_stack_owner

Create the heat_stack_user role

openstack role create heat_stack_user

Installing Heat

Install Components

apt install heat-api heat-api-cfn heat-engine

nano /etc/heat/heat.conf

```
[database]
# ...
connection =
mysql+pymysql://heat:linuxacademy123@controller/heat

[DEFAULT]
transport_url = rabbit://openstack:linuxacademy123@controller
```

Installing Heat

nano /etc/heat/heat.conf

```
[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 = heat
password = linuxacademy123

[trustee]
# ...
auth_type = password
auth_url = http://controller:35357
username = heat
password = linuxacademy123
user_domain_name = default
```

Installing Heat

nano /etc/heat/heat.conf

```
[clients_keystone]
# ...
auth_uri = http://controller:35357

[ec2authtoken]
# ...
auth_uri = http://controller:5000/v3

[DEFAULT]
heat_metadata_server_url = http://controller:8000
heat_waitcondition_server_url =
http://controller:8000/v1/waitcondition

[DEFAULT]
stack_domain_admin = heat_domain_admin
stack_domain_admin_password = linuxacademy123
stack_user_domain_name = heat
```

Installing Heat

Populate the Database

su –s /bin/sh –c “heat-manage db_sync” heat

Finalize Installation

service heat-api restart
service heat-api-cfn restart
service heat-engine restart

Verify Installation

Source Credentials

```
. admin-openrc
```

List Services

```
openstack orchestration service list
```

Deploy and Manage OpenStack Pike on Ubuntu

Launching an Instance

Creating Virtual Networks

Source Admin Credentials

. admin-openrc

Create the Network

openstack network create --share --provider-physical-network provider --provider-network-type flat provider

Creating a Subnet

openstack subnet create --network provider --allocation-pool start=203.0.113.101,end=203.0.113.250 --dns-nameserver 8.8.8.8 --gateway 203.0.113.1 --subnet-range 203.0.113.0/24 provider

Creating a flavor

Create a m1.nano

```
openstack flavor create --id 0 --vcpus 1 --ram 64 --disk 1
m1.nano
```

Generate a key pair

Source the demo credentials

```
. demo-openrc
```

Generate the key pair

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

Upload the key pair

```
openstack keypair create --public-key ~/.ssh/id_rsa.pub mykey
```

Verify the key pair

```
openstack keypair list
```

Create a security group rule

Add rules to default security group

openstack security group rule create --proto icmp default

openstack security group rule create --proto tcp --dst-port 22 default

Review Instance Options

List available flavors

openstack flavor list

List available images

openstack image list

List networks

openstack network list

List security groups

openstack security group list

Launch the Instance

Create the instance

openstack server create --flavor m1.nano --image cirros --nic net-id=PROVIDER_NET_ID --security-group default --key-name mykey provider-instance

Check the status of your instance

openstack server list

Deploy and Manage OpenStack Pike on Ubuntu

Managing Swift ACLs

Controlling Read Access

HTTP referer header can read container contents:

-r ".r:*"

HTTP referer header can read and list container contents:

-r ".r:*,.rlistings"

A list of specific HTTP referer headers permitted to read container contents:

-r ".r:openstack.example.com,.r:swift.example.com"

A list of specific HTTP referer headers denied read access:

-r ".r:*,.r:-openstack.example.com,.r:-swift.example.com"

Controlling Read Access

All users residing in project1 can read container contents:

-r “project1:*”

user1 from project1 can read container contents:

-r “project1:user1”

A list of specific users and projects permitted to read container contents:

-r “project1:user1,project1:user2,project3:*”

Controlling Write Access

All users residing in project1 can write to the container:

-w “project1:*”

user1 from project1 can write to the container:

-w “project1:user1”

A list of specific users and projects permitted to write to the container:

-w “project1:user1,project1:user2,project3:*"

Expiring Objects

Set an object to expire at an absolute time (in Unix time). You can get the current Unix time by running date +'%s'.:

-H "X-Delete-At:UNIX_TIME"

Set an object to expire after a relative amount of time (in seconds):

-H "X-Delete-After:SECONDS"

If you no longer want to expire the object, you can remove the X-Delete-At header:

-H "X-Remove-Delete-At:"

Deploy and Manage OpenStack Pike on Ubuntu

Managing Policies

Example Policies

less /etc/cinder/policy.json

Rules:

```
  "admin_or_owner":  "is_admin:True or (role:admin and
is_admin_project:True) or  project_id:%(project_id)s",
  "default": "rule:admin_or_owner",
  "admin_api": "is_admin:True or (role:admin and
is_admin_project:True)",
```

Policies:

```
  "volume:create": "",
  "volume:create_from_image": "",
  "volume:delete": "rule:admin_or_owner",
  "volume:force_delete": "rule:admin_api",
  "volume:get": "rule:admin_or_owner",
```

Managing Policies

nano /etc/cinder/policy.json

"volume:create": "rule:admin_or_owner",

Restart Service

systemctl restart openstack-cinder-api.service
systemctl restart openstack-cinder-scheduler.service

NOTE: The Lab is running on CentOS, but our installation is Ubuntu. If you want to change policies and restart Cinder on Ubuntu, use the following:
service cinder-scheduler restart
service apache2 restart