Hosting

Openstack

Openstack

OpenStack All?in?One on NVMe + RAID5 SSDs (From Scratch)

This guide is a **from-scratch, manual installation** of an all-in-one OpenStack node on Ubuntu, using:

- **NVMe** for the host OS and control plane services.
- **Three SSDs in RAID 5** (via `mdadm`) for VM ephemeral storage (Nova), block volumes (Cinder via LVM), and optionally images (Glance).

> Target audience: sysadmins and homelab builders who want a reproducible, transparent setup **without DevStack/Packstack**.
> Scope: single host acting as controller + compute + (optional) cinder-volume.
> Production hardening topics are listed at the end.

? Summary (What you'll get)

- **OS**: Ubuntu Server 22.04 LTS
- **OpenStack Release**: Caracal (2024.1) via Ubuntu Cloud Archive
- **Services**: Keystone (Id), Glance (Images), Nova (API/Conductor/Scheduler/NoVNC/Compute), Neutron (ML2 + Linuxbridge), Cinder (LVM backend), Horizon (Dashboard)
- **Storage layout**:
  - `/` on NVMe (host OS)
  - `/dev/md0` = RAID 5 over 3x SSDs ->
    - `cinder-volumes` (LVM VG) for Cinder block storage
    - `/openstack_storage/nova_instances` for Nova ephemeral disks (symlink from `/var/lib/nova/instances`)
    - `/openstack_storage/glance_images` (optional) for Glance filesystem backend

? Assumptions & IP Plan

- Hostname: `controller`

- Management IP: `192.168.0.10/24` (adjust to your LAN)
- DNS/Hosts: you can resolve `controller` to `192.168.0.10`
- Disks:
  - NVMe: OS already installed
  - SSDs: `/dev/sda`, `/dev/sdb`, `/dev/sdc` (adjust device names as needed)
- You have sudo access and outbound internet

> **Note:** If your disk names differ (e.g. `nvme1n1`, `sdb`, `sdc`), adapt commands accordingly.

---

0) Base System Prep

```
sudo apt update && sudo apt -y upgrade
sudo hostnamectl set-hostname controller

# Map controller to IP (edit your values)
echo "192.168.0.10 controller" | sudo tee -a /etc/hosts
```

Install the Cloud Archive for **Caracal (2024.1)** and the OpenStack client:

```
sudo apt install -y software-properties-common
sudo add-apt-repository -y cloud-archive:caracal
sudo apt update && sudo apt -y dist-upgrade
sudo apt install -y python3-openstackclient
```

Reboot if a kernel was upgraded:

```
sudo reboot
```

---

1) Build RAID 5 over the 3 SSDs

Install `mdadm` and create the array:

```
sudo apt install -y mdadm
# Adjust device names to your SSDs
sudo mdadm --create --verbose /dev/md0 --level=5 --raid-devices=3 /dev/sda /dev/sdb /dev/sdc
```

Confirm and monitor sync:

```
cat /proc/mdstat
sudo mdadm --detail /dev/md0
```

Persist the RAID configuration and initramfs:

```
sudo mdadm --detail --scan | sudo tee -a /etc/mdadm/mdadm.conf
sudo update-initramfs -u
```

Create a filesystem and mountpoint (XFS recommended for VM storage):

```
sudo mkfs.xfs /dev/md0
sudo mkdir -p /openstack_storage
sudo blkid /dev/md0
# Copy the UUID and add to /etc/fstab:
# UUID=<your-uuid>  /openstack_storage  xfs  defaults  0  0
sudoedit /etc/fstab
sudo mount -a
df -h | grep openstack_storage
```

---

2) Core Infra: SQL, RabbitMQ, Memcached, NTP

```
# MariaDB (SQL)
sudo apt install -y mariadb-server python3-pymysql

# Harden and bind
sudo mysql_secure_installation
sudo tee /etc/mysql/mariadb.conf.d/99-openstack.cnf >/dev/null <<'EOF'
[mysqld]
bind-address = 0.0.0.0
default-storage-engine = innodb
innodb_file_per_table = on
max_connections = 4096
collation-server = utf8_general_ci
character-set-server = utf8
```

```
EOF

sudo systemctl restart mariadb && sudo systemctl enable mariadb

# RabbitMQ (Messaging)
sudo apt install -y rabbitmq-server
sudo rabbitmqctl add_user openstack StrongRabbitPass!
sudo rabbitmqctl set_permissions openstack ".*" ".*" ".*"

# Memcached (Tokens/Keystone)
sudo apt install -y memcached python3-memcache
sudo sed -i 's/^-l .*/-l 0.0.0.0/' /etc/memcached.conf
sudo systemctl restart memcached && sudo systemctl enable memcached

# NTP (time sync)
sudo apt install -y chrony
sudo systemctl enable --now chrony
```

---

3) Keystone (Identity)

Create DB and user:

```
sudo mysql -u root -p <<'SQL'
CREATE DATABASE keystone;
GRANT ALL PRIVILEGES ON keystone.* TO 'keystone'@'localhost' IDENTIFIED BY
'KEYSTONE_DB_PASS!';
GRANT ALL PRIVILEGES ON keystone.* TO 'keystone'@'%' IDENTIFIED BY 'KEYSTONE_DB_PASS!';
FLUSH PRIVILEGES;
SQL
```

Install & configure Keystone:

```
sudo apt install -y keystone apache2 libapache2-mod-wsgi-py3
sudo sed -i 's|^#* *connection *=.*|connection =
mysql+pymysql://keystone:KEYSTONE_DB_PASS!@controller/keystone|' /etc/keystone/keystone.conf
sudo awk '/^\[token\]/{print;print "provider = fernet";next}1' /etc/keystone/keystone.conf |
sudo tee /etc/keystone/keystone.conf.tmp >/dev/null && sudo mv /etc/keystone/keystone.conf.tmp
/etc/keystone/keystone.conf
```

```
sudo keystone-manage db_sync
sudo keystone-manage fernet_setup --keystone-user keystone --keystone-group keystone
sudo keystone-manage credential_setup --keystone-user keystone --keystone-group keystone

sudo keystone-manage bootstrap \
  --bootstrap-password ADMIN_PASS! \
  --bootstrap-admin-url http://controller:5000/v3/ \
  --bootstrap-internal-url http://controller:5000/v3/ \
  --bootstrap-public-url http://controller:5000/v3/ \
  --bootstrap-region-id RegionOne

sudo systemctl restart apache2 && sudo systemctl enable apache2
```

Admin environment file `~/admin-openrc`:

```
cat > ~/admin-openrc <<'EOF'
export OS_USERNAME=admin
export OS_PASSWORD=ADMIN_PASS!
export OS_PROJECT_NAME=admin
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
EOF

source ~/admin-openrc
openstack token issue
```

Create service project and basic users (glance, nova, placement, neutron, cinder):

```
# Service project
openstack project create --domain Default --description "Service Project" service
```

(Drivers will add their own users in their sections.)

---

4) Glance (Image Service)

DB:

```
sudo mysql -u root -p <<'SQL'
CREATE DATABASE glance;
GRANT ALL PRIVILEGES ON glance.* TO 'glance'@'localhost' IDENTIFIED BY 'GLANCE_DB_PASS!';
GRANT ALL PRIVILEGES ON glance.* TO 'glance'@'%' IDENTIFIED BY 'GLANCE_DB_PASS!';
FLUSH PRIVILEGES;
SQL
```

Identity + install:

```
# Keystone user/role/endpoints
openstack user create --domain Default --password GLANCE_USER_PASS! glance
openstack role add --project service --user glance admin
openstack service create --name glance --description "OpenStack Image" image
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
sudo apt install -y glance
```

Config `/etc/glance/glance-api.conf` (key stanzas):

```
[database]
connection = mysql+pymysql://glance:GLANCE_DB_PASS!@controller/glance

[keystone_authtoken]
www_authenticate_uri = http://controller:5000
auth_url = http://controller:5000
memcached_servers = controller:11211
auth_type = password
project_domain_name = Default
user_domain_name = Default
project_name = service
username = glance
password = GLANCE_USER_PASS!

[glance_store]
stores = file,http
default_store = file
filesystem_store_datadir = /openstack_storage/glance_images/
```

Prepare storage and sync DB:

```
sudo mkdir -p /openstack_storage/glance_images
sudo chown -R glance:glance /openstack_storage/glance_images
sudo glance-manage db_sync
sudo systemctl enable --now glance-api
```

Test image upload (Cirros example):

```
source ~/admin-openrc
wget -O /tmp/cirros.img https://download.cirros-cloud.net/0.6.2/cirros-0.6.2-x86_64-disk.img
openstack image create "cirros" --file /tmp/cirros.img --disk-format qcow2 --container-format
bare --public
openstack image list
```

---

5) Nova (Compute)

DBs + Placement:

```
sudo mysql -u root -p <<'SQL'
CREATE DATABASE nova_api;
CREATE DATABASE nova;
CREATE DATABASE nova_cell0;
CREATE DATABASE placement;
GRANT ALL PRIVILEGES ON nova_api.* TO 'nova'@'localhost' IDENTIFIED BY 'NOVA_DB_PASS!';
GRANT ALL PRIVILEGES ON nova_api.* TO 'nova'@'%' IDENTIFIED BY 'NOVA_DB_PASS!';
GRANT ALL PRIVILEGES ON nova.* TO 'nova'@'localhost' IDENTIFIED BY 'NOVA_DB_PASS!';
GRANT ALL PRIVILEGES ON nova.* TO 'nova'@'%' IDENTIFIED BY 'NOVA_DB_PASS!';
GRANT ALL PRIVILEGES ON nova_cell0.* TO 'nova'@'localhost' IDENTIFIED BY 'NOVA_DB_PASS!';
GRANT ALL PRIVILEGES ON nova_cell0.* TO 'nova'@'%' IDENTIFIED BY 'NOVA_DB_PASS!';
GRANT ALL PRIVILEGES ON placement.* TO 'placement'@'localhost' IDENTIFIED BY
'PLACEMENT_DB_PASS!';
GRANT ALL PRIVILEGES ON placement.* TO 'placement'@'%' IDENTIFIED BY 'PLACEMENT_DB_PASS!';
FLUSH PRIVILEGES;
SQL
```

Keystone users and endpoints:

```
# Nova
openstack user create --domain Default --password NOVA_USER_PASS! nova
openstack role add --project service --user nova admin
openstack service create --name nova --description "OpenStack Compute" compute
openstack endpoint create --region RegionOne compute public   http://controller:8774/v2.1
openstack endpoint create --region RegionOne compute internal http://controller:8774/v2.1
openstack endpoint create --region RegionOne compute admin    http://controller:8774/v2.1

# Placement
openstack user create --domain Default --password PLACEMENT_USER_PASS! placement
openstack role add --project service --user placement admin
openstack service create --name placement --description "Placement API" placement
openstack endpoint create --region RegionOne placement public   http://controller:8778
openstack endpoint create --region RegionOne placement internal http://controller:8778
openstack endpoint create --region RegionOne placement admin    http://controller:8778
```

Install Nova services (API, Conductor, Scheduler, NoVNC) + Placement API:

```
sudo apt install -y nova-api nova-conductor nova-scheduler nova-novncproxy nova-compute \
  placement-api
```

Config highlights:

- `/etc/nova/nova.conf` (minimum):

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

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

[DEFAULT]
transport_url = rabbit://openstack:StrongRabbitPass!@controller
my_ip = 192.168.0.10
use_neutron = True
firewall_driver = nova.virt.firewall.NoopFirewallDriver
enabled_apis = osapi_compute,metadata

[keystone_authtoken]
www_authenticate_uri = http://controller:5000
```

```
auth_url = http://controller:5000
memcached_servers = controller:11211
auth_type = password
project_domain_name = Default
user_domain_name = Default
project_name = service
username = nova
password = NOVA_USER_PASS!

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

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

[placement]
region_name = RegionOne
project_domain_name = Default
project_name = service
auth_type = password
user_domain_name = Default
auth_url = http://controller:5000/v3
username = placement
password = PLACEMENT_USER_PASS!
```

- `/etc/placement/placement.conf`:

```
[placement_database]
connection = mysql+pymysql://placement:PLACEMENT_DB_PASS!@controller/placement

[api]
auth_strategy = keystone

[keystone_authtoken]
auth_url = http://controller:5000/v3
memcached_servers = controller:11211
auth_type = password
project_domain_name = Default
user_domain_name = Default
project_name = service
```

```
username = placement
password = PLACEMENT_USER_PASS!
```

DB sync + cell mapping:

```
sudo su -s /bin/bash nova -c "nova-manage api_db sync"
sudo su -s /bin/bash nova -c "nova-manage cell_v2 map_cell0"
sudo su -s /bin/bash nova -c "nova-manage cell_v2 create_cell --name=cell1"
sudo su -s /bin/bash nova -c "nova-manage db sync"
```

Restart services:

```
sudo systemctl restart placement-api
sudo systemctl restart nova-api nova-scheduler nova-conductor nova-novncproxy
sudo systemctl restart nova-compute
sudo systemctl enable placement-api nova-api nova-scheduler nova-conductor nova-novncproxy
nova-compute
```

Move Nova instance storage to RAID

```
sudo systemctl stop nova-compute
sudo mkdir -p /openstack_storage/nova_instances
sudo rsync -aHAX /var/lib/nova/instances/ /openstack_storage/nova_instances/
sudo mv /var/lib/nova/instances /var/lib/nova/instances.bak
sudo ln -s /openstack_storage/nova_instances /var/lib/nova/instances
sudo systemctl start nova-compute
```

---

6) Neutron (Networking) — ML2 + Linuxbridge (simple flat/VLAN)

Install:

```
sudo apt install -y neutron-server neutron-plugin-ml2 neutron-linuxbridge-agent neutron-dhcp-
agent neutron-metadata-agent neutron-l3-agent
```

Config `/etc/neutron/neutron.conf` (highlights):

```
[DEFAULT]
core_plugin = ml2
service_plugins = router
transport_url = rabbit://openstack:StrongRabbitPass!@controller
auth_strategy = keystone
notify_nova_on_port_status_changes = true
notify_nova_on_port_data_changes = true

[keystone_authtoken]
www_authenticate_uri = http://controller:5000
auth_url = http://controller:5000
memcached_servers = controller:11211
auth_type = password
project_domain_name = Default
user_domain_name = Default
project_name = service
username = neutron
password = NEUTRON_USER_PASS!

[nova]
auth_url = http://controller:5000
auth_type = password
project_domain_name = Default
user_domain_name = Default
region_name = RegionOne
project_name = service
username = nova
password = NOVA_USER_PASS!
```

Create Neutron user/endpoints:

```
openstack user create --domain Default --password NEUTRON_USER_PASS! neutron
openstack role add --project service --user neutron admin
openstack service create --name neutron --description "OpenStack Networking" network
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
```

ML2 and Linuxbridge basics:

- `/etc/neutron/plugins/ml2/ml2_conf.ini` (flat or VLAN example):

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

[ml2_type_flat]
flat_networks = physnet1

[securitygroup]
enable_ipset = true
```

- `/etc/neutron/plugins/ml2/linuxbridge_agent.ini`:

```
[linux_bridge]
physical_interface_mappings = physnet1:enp3s0

[vxlan]
enable_vxlan = false

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

- Metadata agent `/etc/neutron/metadata_agent.ini`:

```
[DEFAULT]
nova_metadata_host = controller
metadata_proxy_shared_secret = METADATA_SECRET!
```

- In Nova `/etc/nova/nova.conf`, add metadata secret:

```
[neutron]
auth_url = http://controller:5000
auth_type = password
project_domain_name = Default
user_domain_name = Default
region_name = RegionOne
project_name = service
username = neutron
```

```
password = NEUTRON_USER_PASS!
metadata_proxy_shared_secret = METADATA_SECRET!
service_metadata_proxy = True
```

Restart:

```
sudo ln -sf /etc/neutron/plugins/ml2/ml2_conf.ini /etc/neutron/plugin.ini
sudo systemctl restart nova-api
sudo systemctl restart neutron-server neutron-linuxbridge-agent neutron-dhcp-agent neutron-metadata-agent neutron-l3-agent
sudo systemctl enable neutron-server neutron-linuxbridge-agent neutron-dhcp-agent neutron-metadata-agent neutron-l3-agent
```

Create a provider network (flat) and subnet (example):

```
source ~/admin-openrc
openstack network create --share --external --provider-physical-network physnet1 --provider-network-type flat public
openstack subnet create --network public --allocation-pool
start=192.168.0.200,end=192.168.0.250 \
  --dns-nameserver 1.1.1.1 --gateway 192.168.0.1 --subnet-range 192.168.0.0/24 public-subnet
```

---

7) Cinder (Block Storage) — LVM on RAID5

Create LVM PV + VG on `/dev/md0`:

```
sudo apt install -y lvm2
sudo pvcreate /dev/md0
sudo vgcreate cinder-volumes /dev/md0
```

Install Cinder services:

```
sudo apt install -y cinder-api cinder-scheduler cinder-volume tgt
```

DB + Keystone:

```
sudo mysql -u root -p <<'SQL'
CREATE DATABASE cinder;
GRANT ALL PRIVILEGES ON cinder.* TO 'cinder'@'localhost' IDENTIFIED BY 'CINDER_DB_PASS!';
GRANT ALL PRIVILEGES ON cinder.* TO 'cinder'@'%' IDENTIFIED BY 'CINDER_DB_PASS!';
FLUSH PRIVILEGES;
SQL

openstack user create --domain Default --password CINDER_USER_PASS! cinder
openstack role add --project service --user cinder admin
openstack service create --name cinder --description "OpenStack Block Storage" volumev3
openstack endpoint create --region RegionOne volumev3 public
http://controller:8776/v3/%\(project_id\)s
openstack endpoint create --region RegionOne volumev3 internal
http://controller:8776/v3/%\(project_id\)s
openstack endpoint create --region RegionOne volumev3 admin
http://controller:8776/v3/%\(project_id\)s
```

Key config `/etc/cinder/cinder.conf`:

```
[database]
connection = mysql+pymysql://cinder:CINDER_DB_PASS!@controller/cinder

[DEFAULT]
transport_url = rabbit://openstack:StrongRabbitPass!@controller
auth_strategy = keystone
my_ip = 192.168.0.10
enabled_backends = lvm
glance_api_servers = http://controller:9292

[keystone_authtoken]
www_authenticate_uri = http://controller:5000
auth_url = http://controller:5000
memcached_servers = controller:11211
auth_type = password
project_domain_name = Default
user_domain_name = Default
project_name = service
username = cinder
password = CINDER_USER_PASS!
```

```
[lvm]
volume_driver = cinder.volume.drivers.lvm.LVMVolumeDriver
volume_group = cinder-volumes
target_protocol = iscsi
target_helper = tgtadm
volume_backend_name = lvm
```

DB sync + restart:

```
sudo su -s /bin/bash cinder -c "cinder-manage db sync"
sudo systemctl restart cinder-scheduler cinder-volume cinder-api tgt
sudo systemctl enable cinder-scheduler cinder-volume cinder-api tgt
```

Verify:

```
openstack volume service list
```

---

8) Horizon (Dashboard)

```
sudo apt install -y openstack-dashboard
sudo sed -i 's/^OPENSTACK_HOST.*/OPENSTACK_HOST = "controller"/' /etc/openstack-
dashboard/local_settings.py
# (Optional) allow all hosts in dev/lab:
sudo sed -i 's/^ALLOWED_HOSTS.*/ALLOWED_HOSTS = ["*"]/' /etc/openstack-
dashboard/local_settings.py
sudo systemctl restart apache2
```

Browse to: `http://controller/horizon` and log in as `admin` with `ADMIN_PASS!`

---

9) Quick Functional Test

```
source ~/admin-openrc
openstack image list
openstack network list

# Create minimal flavor & keypair
```

```
openstack flavor create --ram 512 --disk 1 --vcpus 1 m1.tiny
openstack keypair create --public-key ~/.ssh/id_rsa.pub mykey

# Boot cirros on provider network (adjust net ID/name)
NET_ID=$(openstack network show public -f value -c id)
openstack server create --flavor m1.tiny --image cirros --nic net-id=$NET_ID --key-name mykey
testvm1

openstack server list
```

Use the dashboard or CLI to access the console (NoVNC) and verify boot.

---

10) RAID Health, Monitoring & Benchmark

Health:

```
cat /proc/mdstat
sudo mdadm --detail /dev/md0
```

Email alerts (optional):

```
sudo apt install -y postfix mailutils
sudo mdadm --monitor --scan --mail=root@localhost --delay=300
```

Benchmark (optional, `fio`):

```
sudo apt install -y fio
fio --name=raid5test --filename=/openstack_storage/testfile --size=2G --rw=readwrite --bs=1M
--numjobs=4 --direct=1 --group_reporting
```

Hardening & Production Notes (Read Me)

- Replace all `*_PASS!` with strong secrets; store in a password manager.

- Use a **separate data NIC** and bridges for provider/tenant traffic.
- Consider **BTRFS/ZFS** if you want snapshots/CRC with different trade-offs (not covered here).
- For Cinder at scale, prefer **iSCSI/NFS backends** (e.g., LVM over dedicated disks, Ceph for HA).
- Enable **TLS** for Keystone/Glance/Nova/Neutron/Cinder endpoints.
- Configure **backups** for MariaDB and `/etc/` configs, and take regular **etcd/RabbitMQ** backups if applicable.
- Put OpenStack services under **systemd overrides** and logrotate tuning.
- Use **Kolla-Ansible** or **OpenStack-Ansible** for multi-node, HA, and day-2 ops once you outgrow single-node.

Troubleshooting Quickies

- `openstack service list` / `openstack endpoint list` to confirm identity layer.
- `journalctl -u <service> -e` for any failing service.
- Verify DB connectivity with `mysql -h controller -u <user> -p <db>`.
- Placement errors? Check `/etc/nova/nova.conf` `[placement]` and restart `placement-api` and `nova-*`.
- Metadata 404? Confirm `metadata_proxy_shared_secret` matches in Neutron metadata agent and Nova `[neutron]` section.
- No external connectivity for VMs? Revisit provider network mapping and Linuxbridge `physical_interface_mappings`.

Appendix: Variable Cheat Sheet

Replace these with your own strong values:

```
ADMIN_PASS!
KEYSTONE_DB_PASS!
GLANCE_DB_PASS!
GLANCE_USER_PASS!
NOVA_DB_PASS!
PLACEMENT_DB_PASS!
PLACEMENT_USER_PASS!
NEUTRON_USER_PASS!
CINDER_DB_PASS!
CINDER_USER_PASS!
METADATA_SECRET!
```

```
StrongRabbitPass!
```

You’re set. You now have an all-in-one OpenStack host with **NVMe OS** and **RAID5 SSD-backed storage** for instances, volumes, and images. Happy hacking!