OPENSTACK BASIC

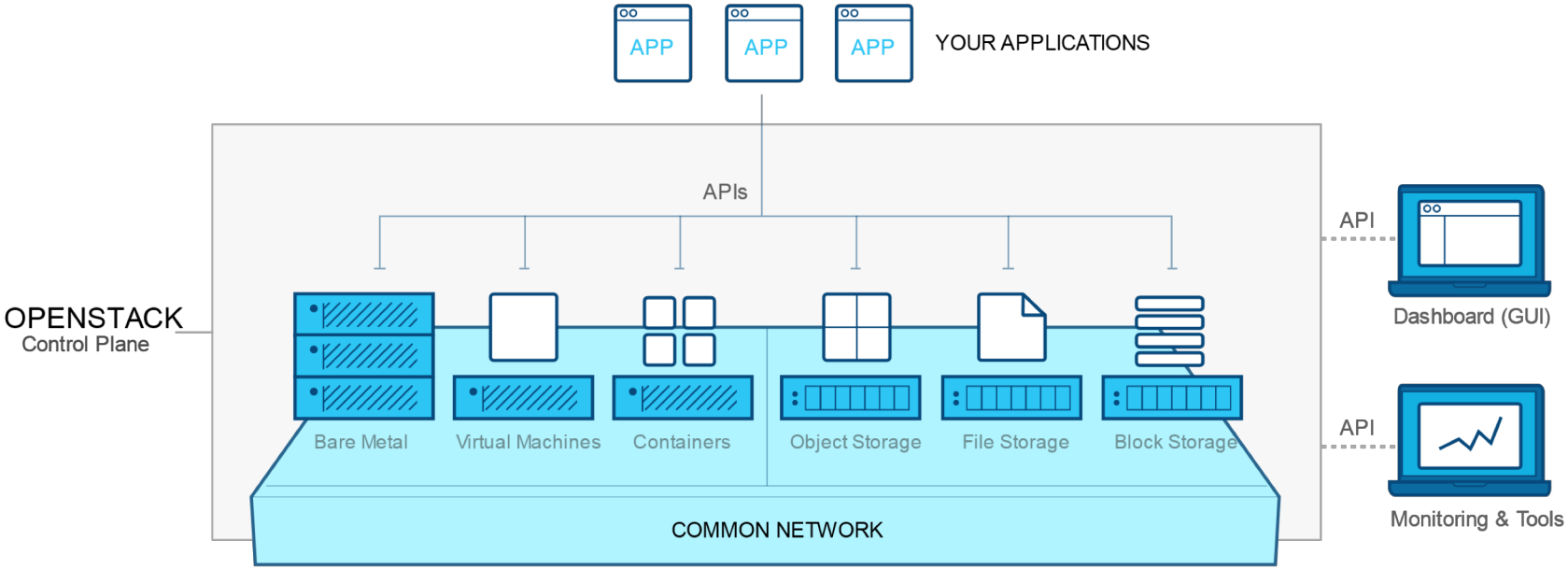
Forked from Wey Gu's Learning Openstack from scratch for junior telco. engineers

OVERVIEW

- Introduction
  - AWS, IaaS, VIM, NFVi, HA
- Components/Sub Projects
  - Nova, Neutron, Cinder, Glance, Keystone
- Some Backend Techs
  - KVM, OVS, Linux Bridge, DPDK, HaProxy, RabbitMQ, Python, MySQL, Pacemaker, iSCSI, network namespace
- Communication between them
  - RPC, RESTful
- Installation of main components
  - Keystone, Glance, Cinder, Nova, Neutron
- Openstack Operation
  - Dashboard, Cli, RESTful API
- Inner flows
  - Nova:Request Flow of booting VM, Neutron:how did package go, cinder: provisioning a block(iscsi backend)
- NFV
  - Orchestration, Lifecycle management, VNF, Building a vApp from VNFD to lifecycle
- Troubleshooting
  - Wireshark trace, log, Python(exceptions), oslo.message, debug
- Build a tiny hand-made ‘ECM’

WHAT IS OPENSTACK

- IaaS solution
- AWS-like software solution
- Opensource Project



WHY OPENSTACK

- Massively scalable and elastic, enabling support for large volumes of data and large numbers of users.
- Open source, with a large, active community contributing to frequent updates and an ever-expanding set of features.
- Ideal for big data, analytics, mobile applications, grid computing, and other leading use cases.
- Designed to run on commodity hardware, driving cost and flexibility benefits.

PROJECTS IN OPENSTACK

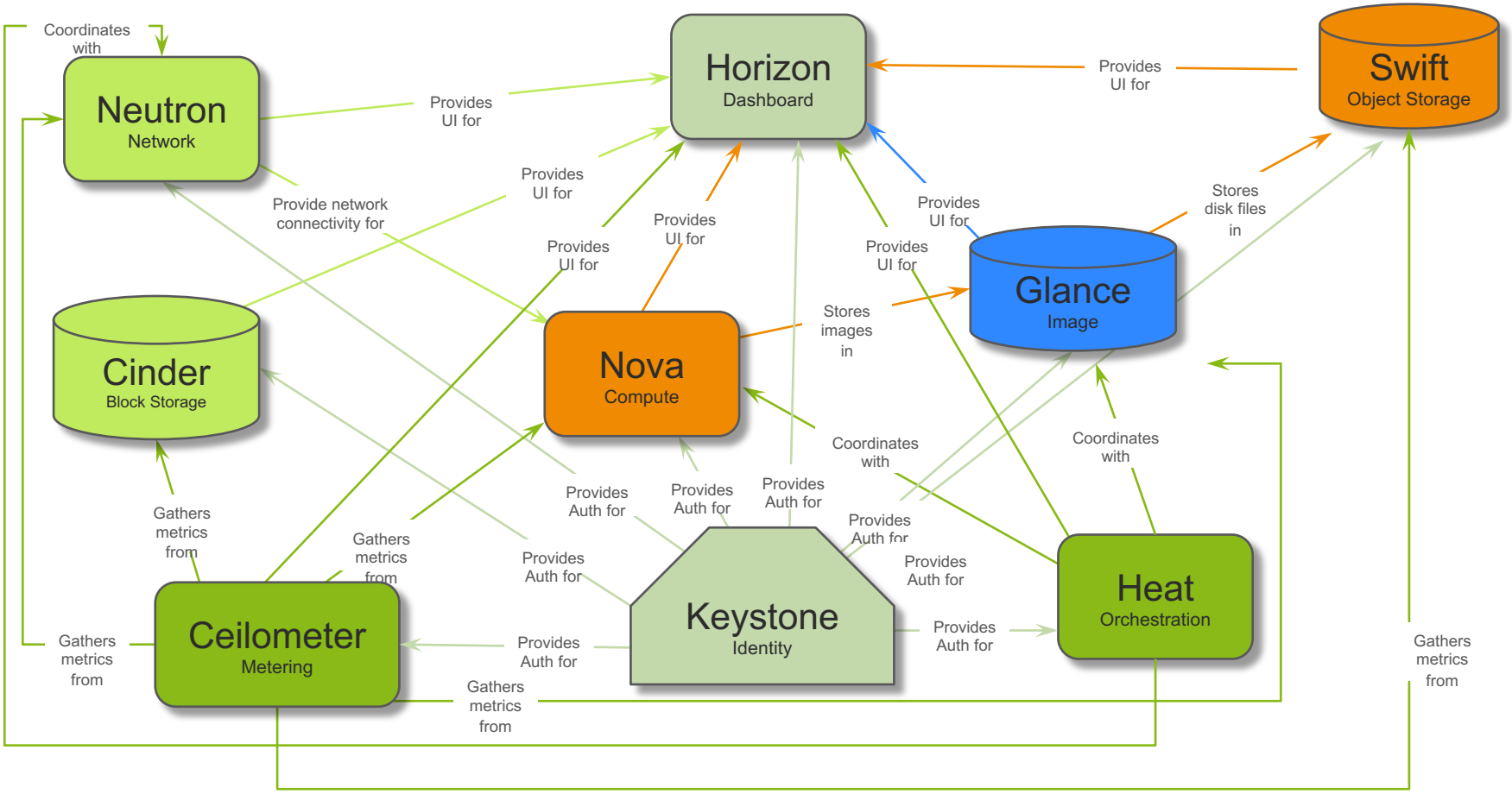
Core Projects

- Nova, Compute
- Neutron, Networking
- Swift, Object Storage
- Keystone, Identity Service
- Cinder, Block Storage

Some other Projects

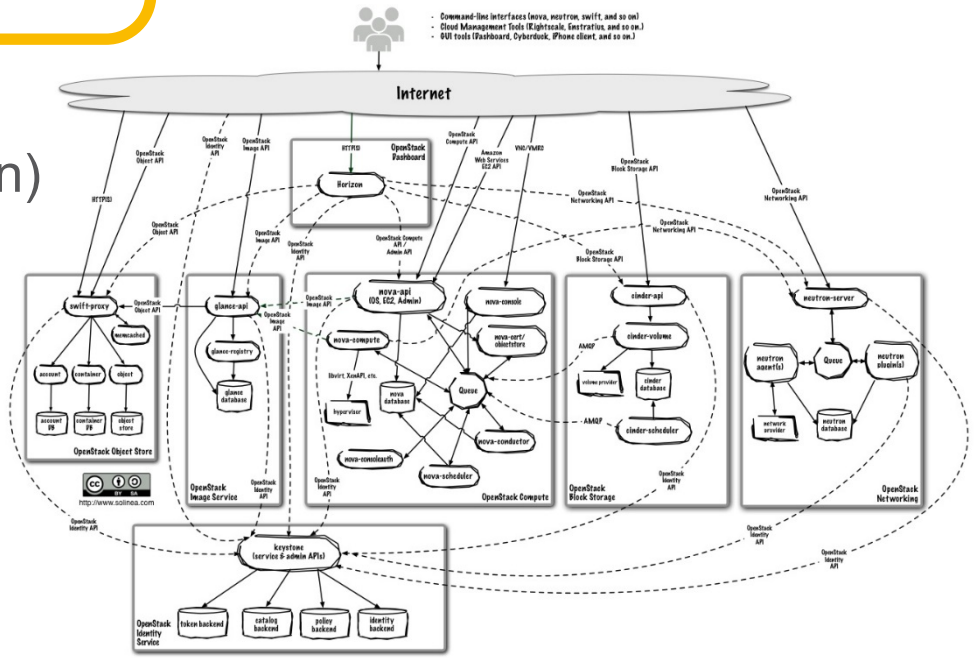
- Heat, Orchestration
- Horizon, Dashboard
- Rally, Benchmark
- Ceilometer, Metering & Data Collection Service
- Aodh, Alarming Service

HOW THEY CO-WORK?



HOW: COMMUNICATE

- RESTful API
  - WSGI by paste/pescan (Python)
    - http/https based
- RPC call/ Notification
  - Via oslo.messaging (Python)
    - AMQP
      - RabbitMQ (Erlang)
      - Kafka



DOMO PYTHON WEB SERVICE (WSGI)

› A piece of code to do web service like WSGI

```
# -*- coding: utf-8 -*-
#!/usr/bin/python
#This is the demo WSGI.py, imagin it's NOVA-API :-p
from flask import Flask
app = Flask(__name__)
@app.route("/hello")
def hello():
    #API GET /hello
    #Do nothing, just return a string.
    return "Hello World! I am a WSGI process :-)..."
```

› Demo more codes

Fetch latest demo code from:

https://github.com/littlewey/workshops/tree/master/00-Openstack-Basic/demo-wsgi

RESTFUL API

- REST (or Representational State Transfer)
  - is a way of writing stateless client/server applications – in other words, a software architecture style. In short, the most common way to implement a RESTful service is to use HTTP as the underlying transport, and if you do, every service request is an HTTP method, such as HTTP GET/PUT/POST/DELETE/PATCH etc.
  - REST has overtaken other styles of Web service development, such SOAP or WSDL, primarily because of its ability to provide reliable, scalable, secure, and easy to use and integrate Web services.
  - Understanding the details of REST is important if you are an OpenStack developer so you can create such a service, but not so important if you want to use the OpenStack services.
  - As a user you want to find the best way to consume the OpenStack API, and leave the complexity of REST to the producers of the API.

EXAMPLE OF USING RESTFUL API (1/2)

A Blog web service RESTful API call example

```
$ curl -g -i -X GET http://jsonplaceholder.typicode.com/posts/32

HTTP/1.1 200 OK
Date: Sat, 26 Aug 2017 13:13:42 GMT
Content-Type: application/json; charset=utf-8
Content-Length: 292
{
  "userId": 1,
  "id": 32,
  "title": "What a beautiful Day",
  "body" : "Today I went for surfing at Dalian Laohutan Beach."
}
```

Demo Other API calls

```
$ curl -g -i -X GET http://jsonplaceholder.typicode.com/posts
$ curl -g -i -X DELETE http://jsonplaceholder.typicode.com/posts/2
```

EXAMPLE OF USING RESTFUL API (2/2)

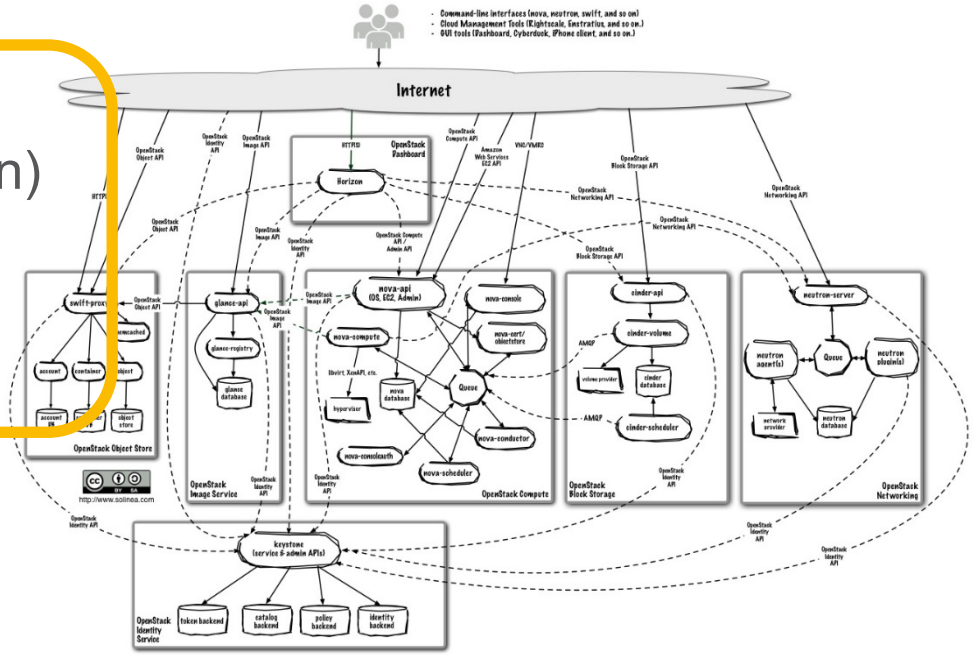
```
$ curl -v -s -X POST $OS_AUTH_URL/auth/tokens?nocatalog \
 -H "Content-Type: application/json"   \
 -d '{ "auth": { "identity": { "methods": ["password"],"password": {"user": {"domain": {"n
"domain": { "name": "'"$OS_PROJECT_DOMAIN_NAME"'" }, "name":  "'"$OS_PROJECT_NAME"'" } } }

…
> POST /v3/auth/tokens?nocatalog HTTP/1.1
> Host: controller:35357
> User-Agent: curl/7.47.0
…
< X-Subject-Token: gAAAAABZoXbwgup7RP5gs6mTIXfo3tw5xo8y5OGuIA7fT0wTFxqgatWu5MopCVxfmPQMrhp
…
$ export OS_TOKEN=<X-Subject-Token>
$ curl -s -H "X-Auth-Token: $OS_TOKEN" http://controller:8774/v2.1/flavors |python -m jso

{
    "flavors": [
        {
            "id": "0",
            "links": [
                {
                    "href": "http://controller:8774/v2.1/flavors/0",
                    "rel": "self"
                },
```

HOW: COMMUNICATE

- RESTful API
  - WSGI by paste/pescan (Python)
    - http/https based
- RPC call/ Notification
  - Via oslo.messaging (Python)
    - AMQP
      - RabbitMQ (Erlang)
      - Kafka



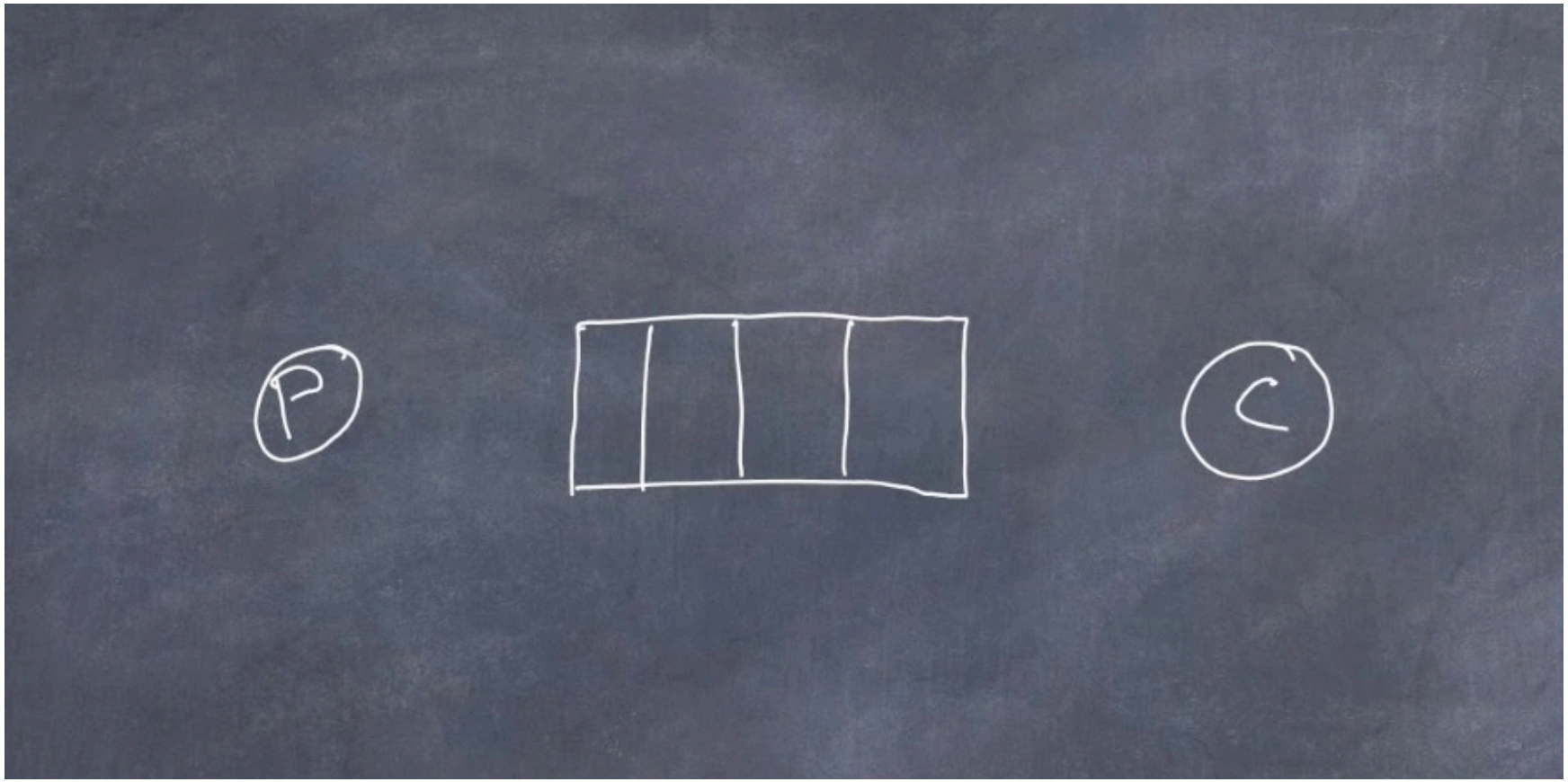
RPC VIA AMQP

- Reliable
- Efficient

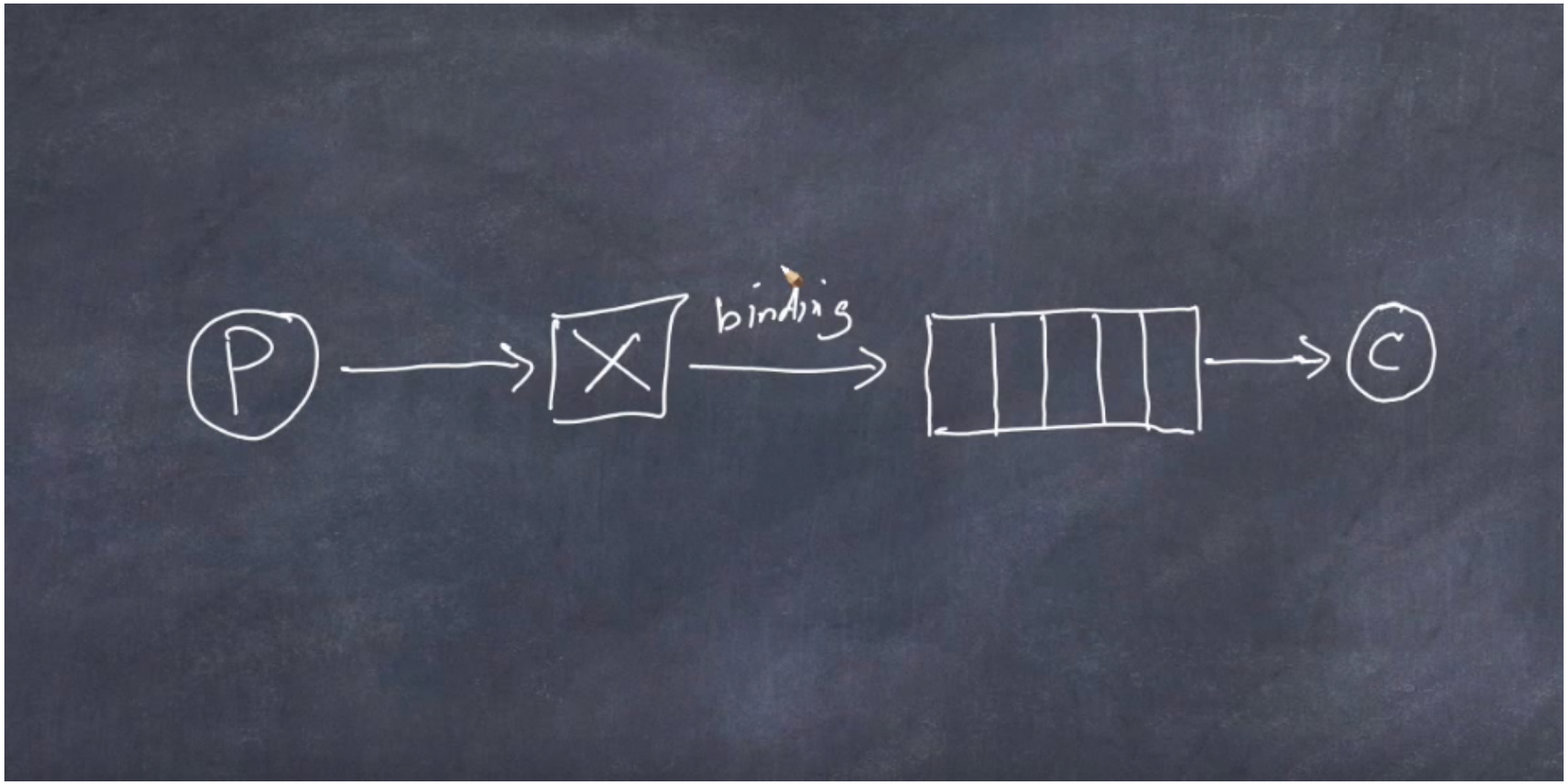
OSLO.MESSAGING

- building RPC clients/servers
  - RPC, User await for response, client secures it.
- emitting/handling Notification
  - User cast messages and don’t expect response, backend like Rabbit-MQ secures it reaching target (Kafka may be a better backend for event notification)

AMQP & RABBITMQ



AMQP & RABBITMQ



AMQP & RABBITMQ

- Message broker, provide exchange and Queue for Publisher and Consumer
- Type:
  - Fanout
  - Direct
  - Headers
  - Topic
- Feature:
  - Can be reliable (P got confirm from B, B got acknowledge from C)
  - C deal with duplicated
  - B persistently saves msg

RPC CALL VIA RABBITMQ FLOW

- Flow
  1. Client -> request -> RabbitMQ
  2. RabbitMQ -> request -> Server
  3. Server processes the request and produces the response
  4. Server -> response -> RabbitMQ
  5. RabbitMQ -> response -> Client

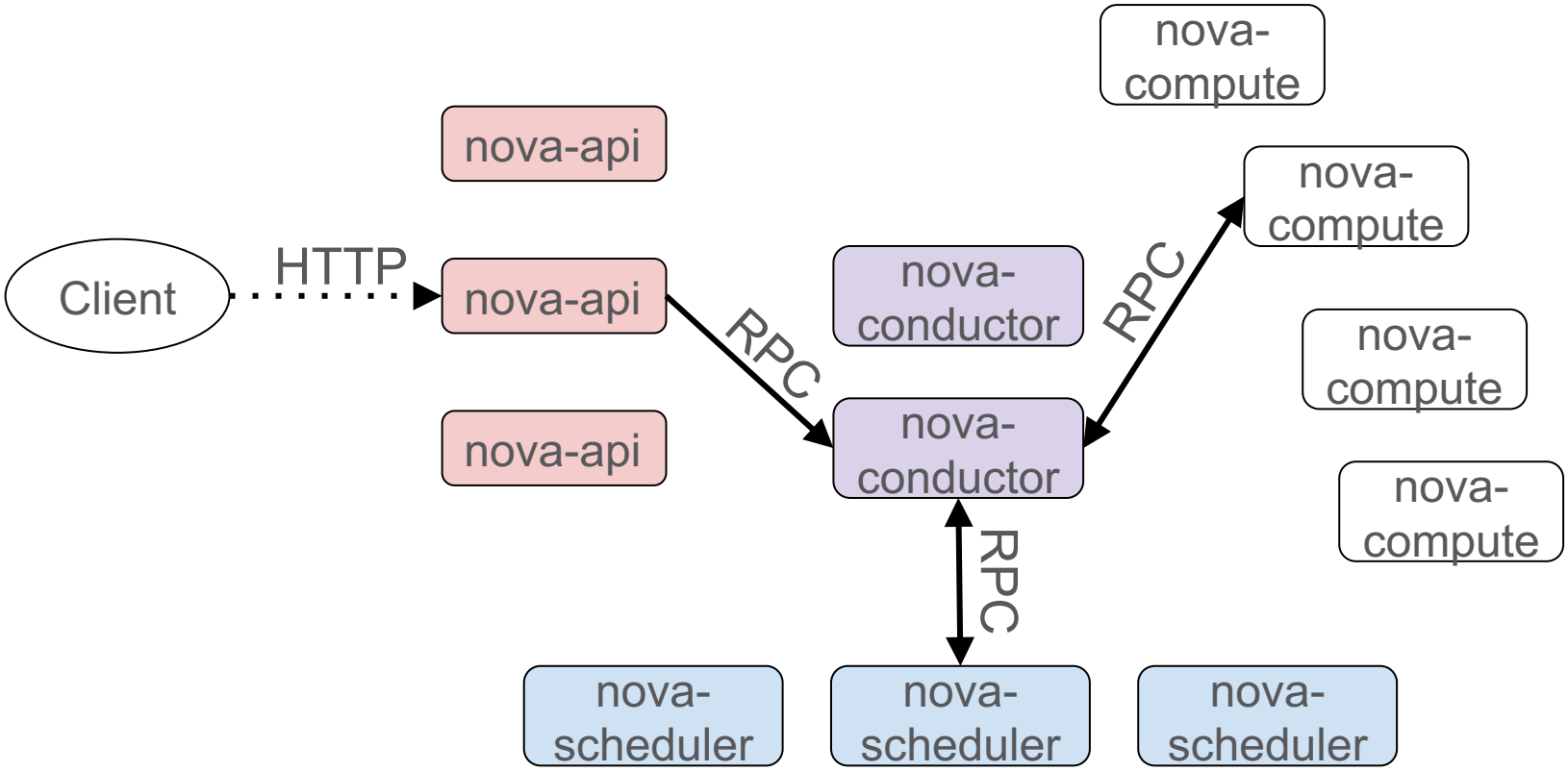
- Typical error:
  - If the process gets stuck on *any* step from 2 to 5, client gets a MessagingTimeout exception.

Note: client, server here is for RPC not rabbitMQ

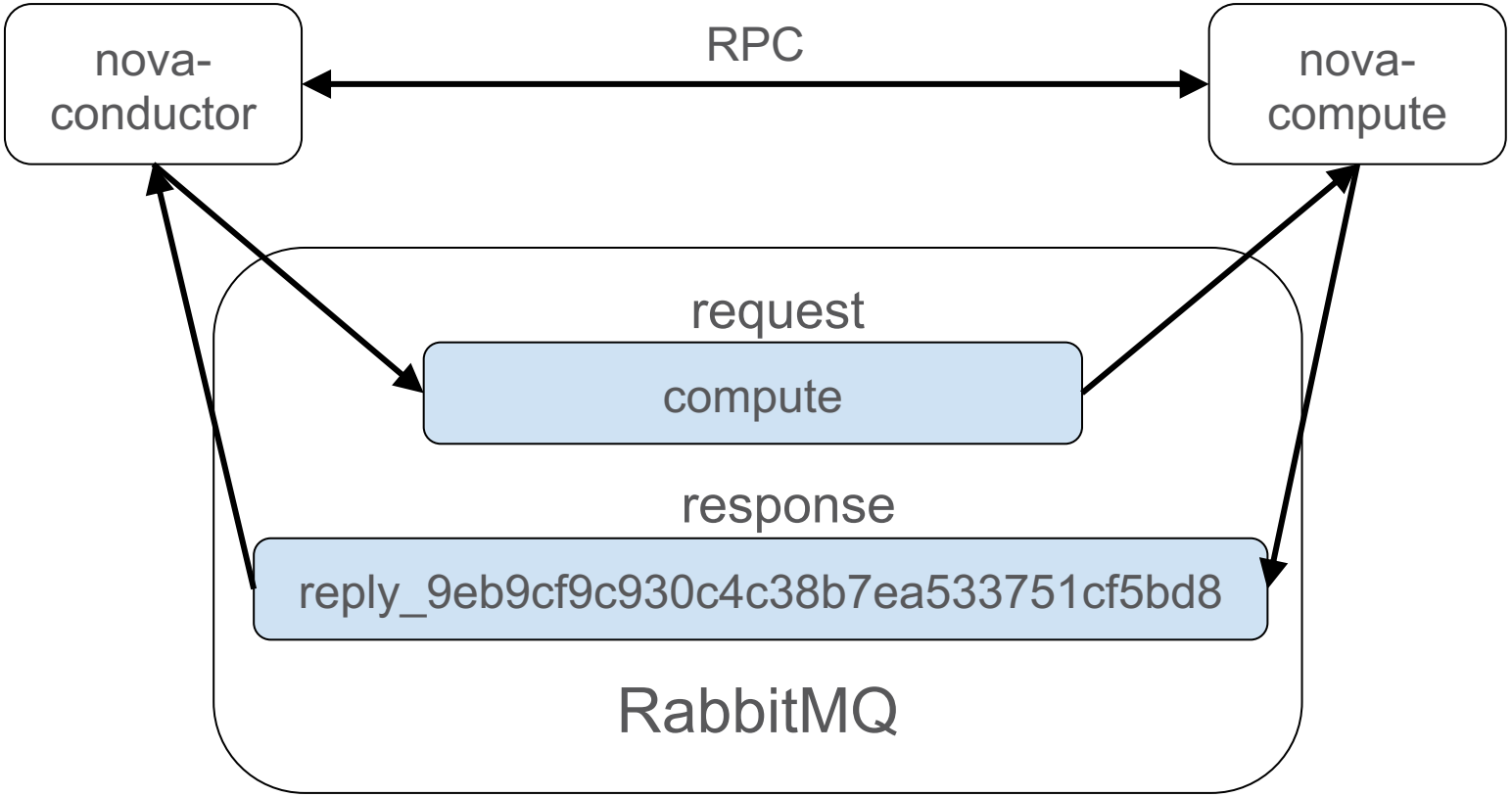
AN ERROR LOG OF MSG TIMEOUT IN NOVA COMPUTE

```
[req-ab05c74c-e56c-408e-b302-4a87bb9e68e7 - - - - -] Error during ComputeManager.update_avail
Traceback (most recent call last):
  File "/usr/lib/python2.7/dist-packages/oslo_service/periodic_task.py", line 220, in run_per
    task(self, context)
  File "/usr/lib/python2.7/dist-packages/nova/compute/manager.py", line 6592, in update_avail
    use_slave=True)
  File "/usr/lib/python2.7/dist-packages/nova/compute/manager.py", line 6615, in _get_compute
    use_slave=use_slave)
  File "/usr/lib/python2.7/dist-packages/oslo_versionedobjects/base.py", line 177, in wrapper
    args, kwargs)
  File "/usr/lib/python2.7/dist-packages/nova/conductor/rpcapi.py", line 239, in object_class
    args=args, kwargs=kwargs)
  File "/usr/lib/python2.7/dist-packages/oslo_messaging/rpc/client.py", line 169, in call
    retry=self.retry)
  File "/usr/lib/python2.7/dist-packages/oslo_messaging/transport.py", line 97, in _send
    timeout=timeout, retry=retry)
  …
    message = self.waiters.get(msg_id, timeout=timeout)
  File "/usr/lib/python2.7/dist-packages/oslo_messaging/_drivers/amqpdriver.py", line 238, in
    'to message ID %s' % msg_id)
    MessagingTimeout: Timed out waiting for a reply to message ID cc8c02f1714a45a290733d3abf
```

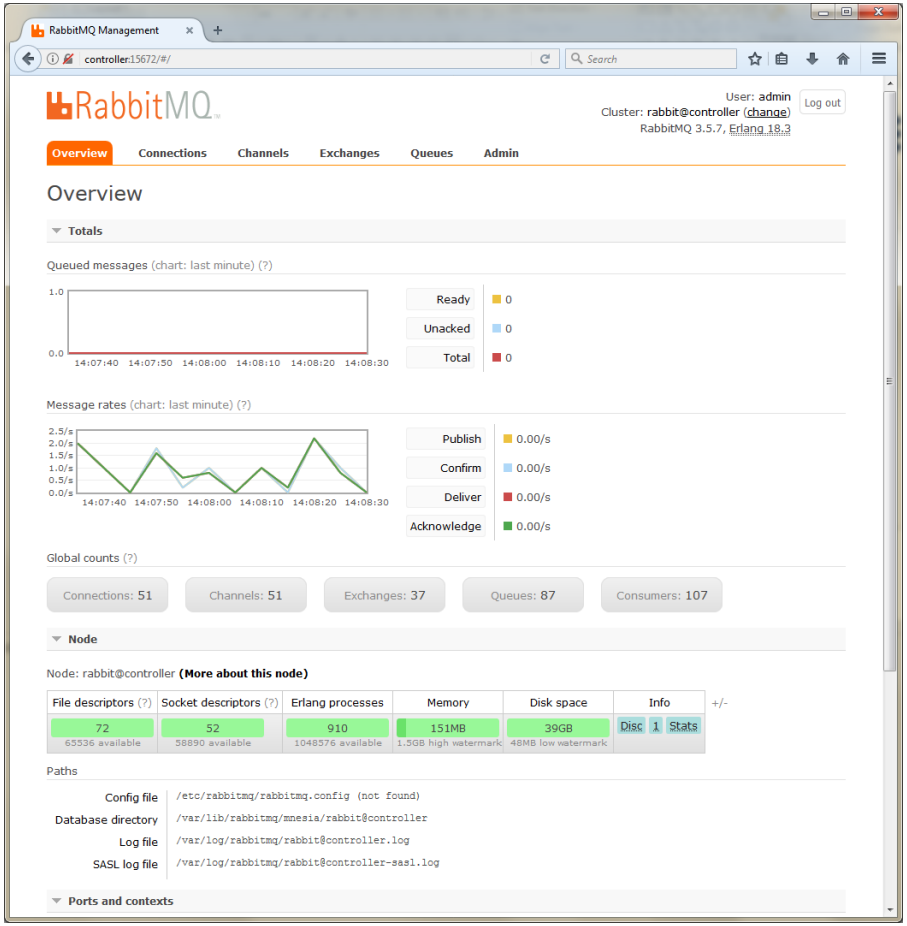
SPAWNING A VM IN NOVA

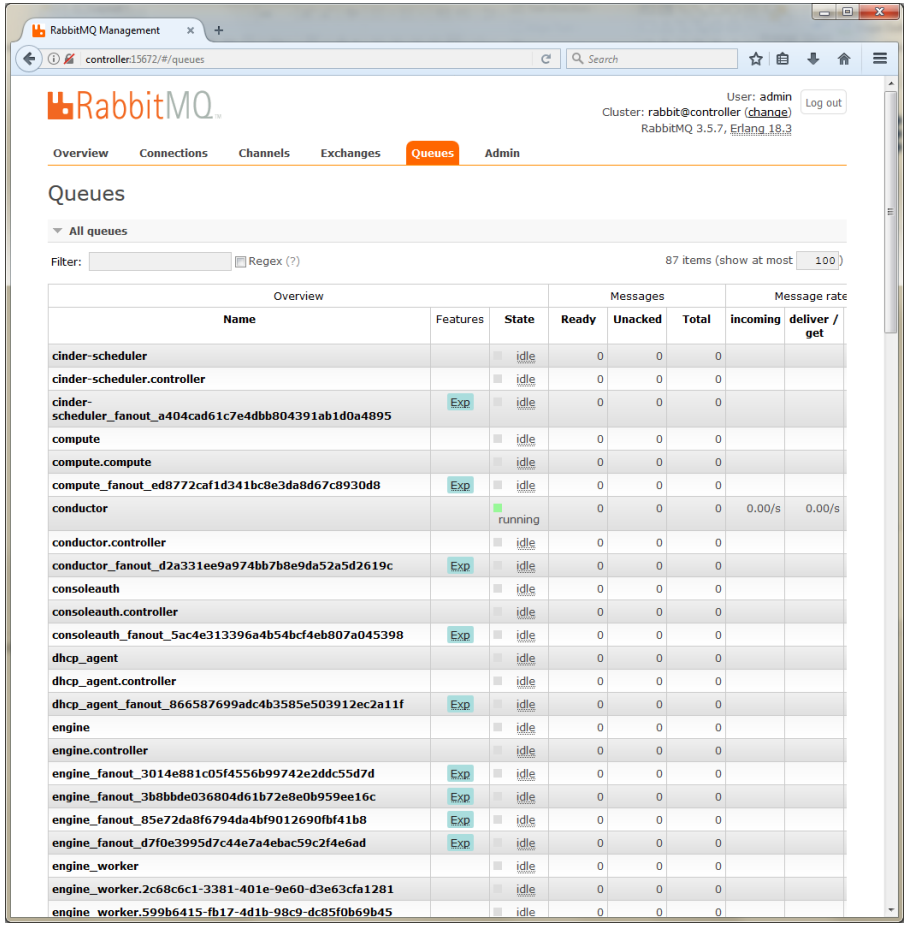


WHERE IS RABBITMQ IN THIS PICTURE?



CHECK ON RABBITMQ GUI





TROUBLESHOOTING OSLO.MESSING

› Debug log mode

```
[DEFAULT]
# debug=true
default_log_levels=...,oslo.messaging=DEBUG,...
```

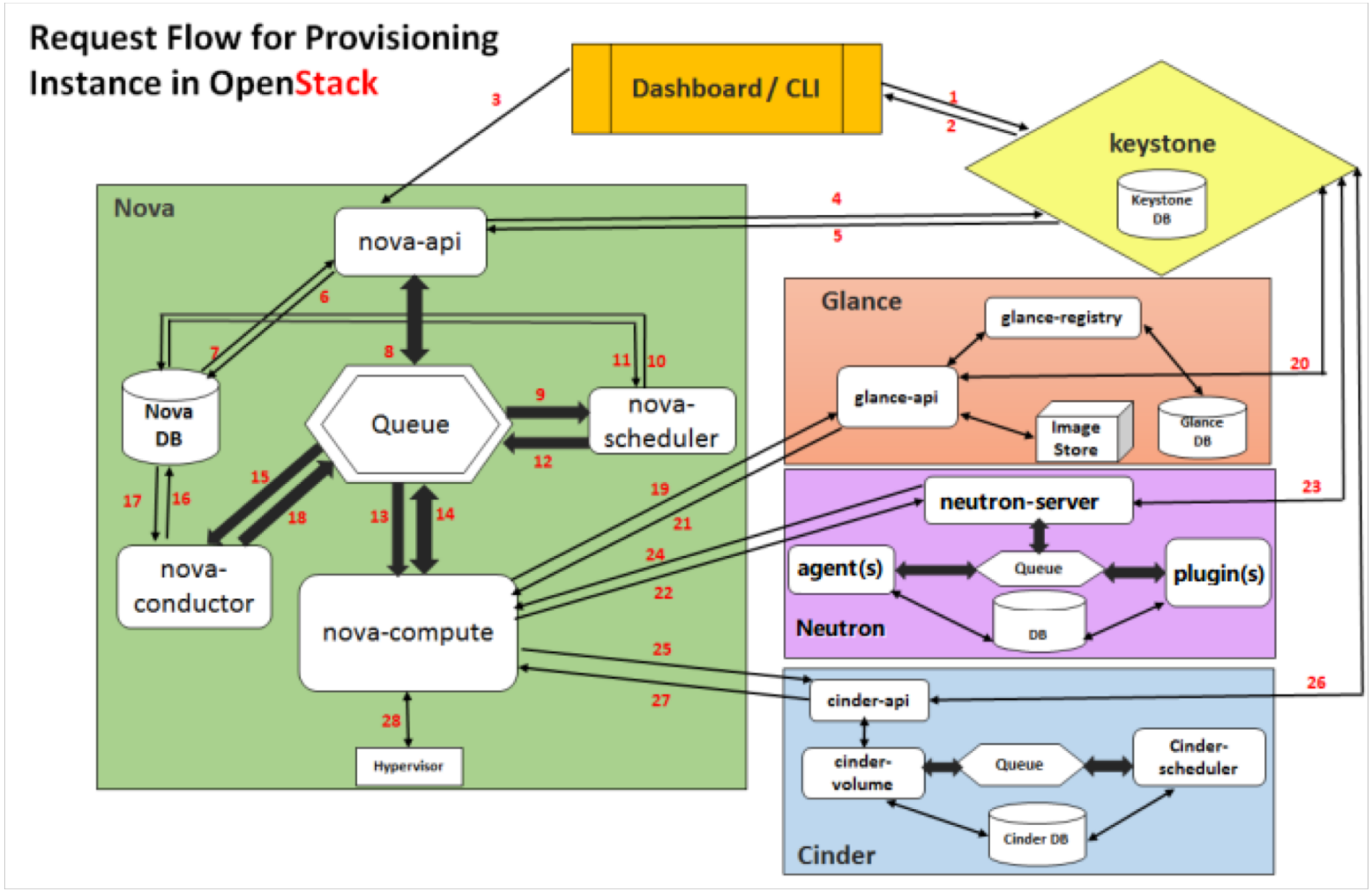
› Commands

```
$ rabbitmqctl --help
$ rabbitmqctl list_queues / list_exchanges / … / status / cluster_status
$ rabbitmqctl trace_on -p my-vhost
$ rabbitmqctl list_queues  memory consumers messages name
```

› Reference :

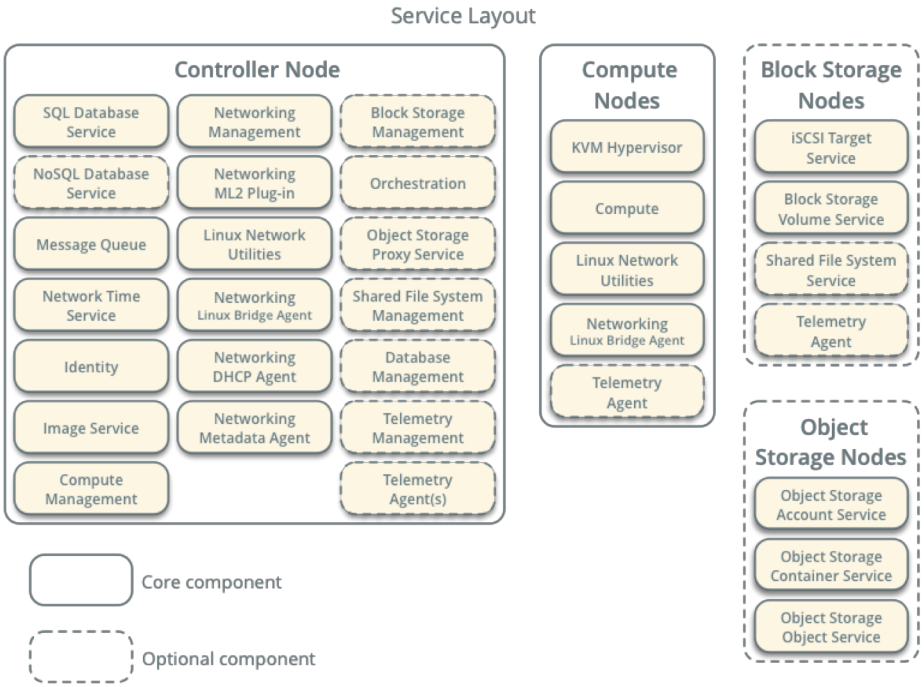
- Austin Summit - oslo.messaging issues.pptx

HOW: CALL FLOW BOOTING VM AKA A NOVA INSTANCE



HOW: DEPLOYED IN HW

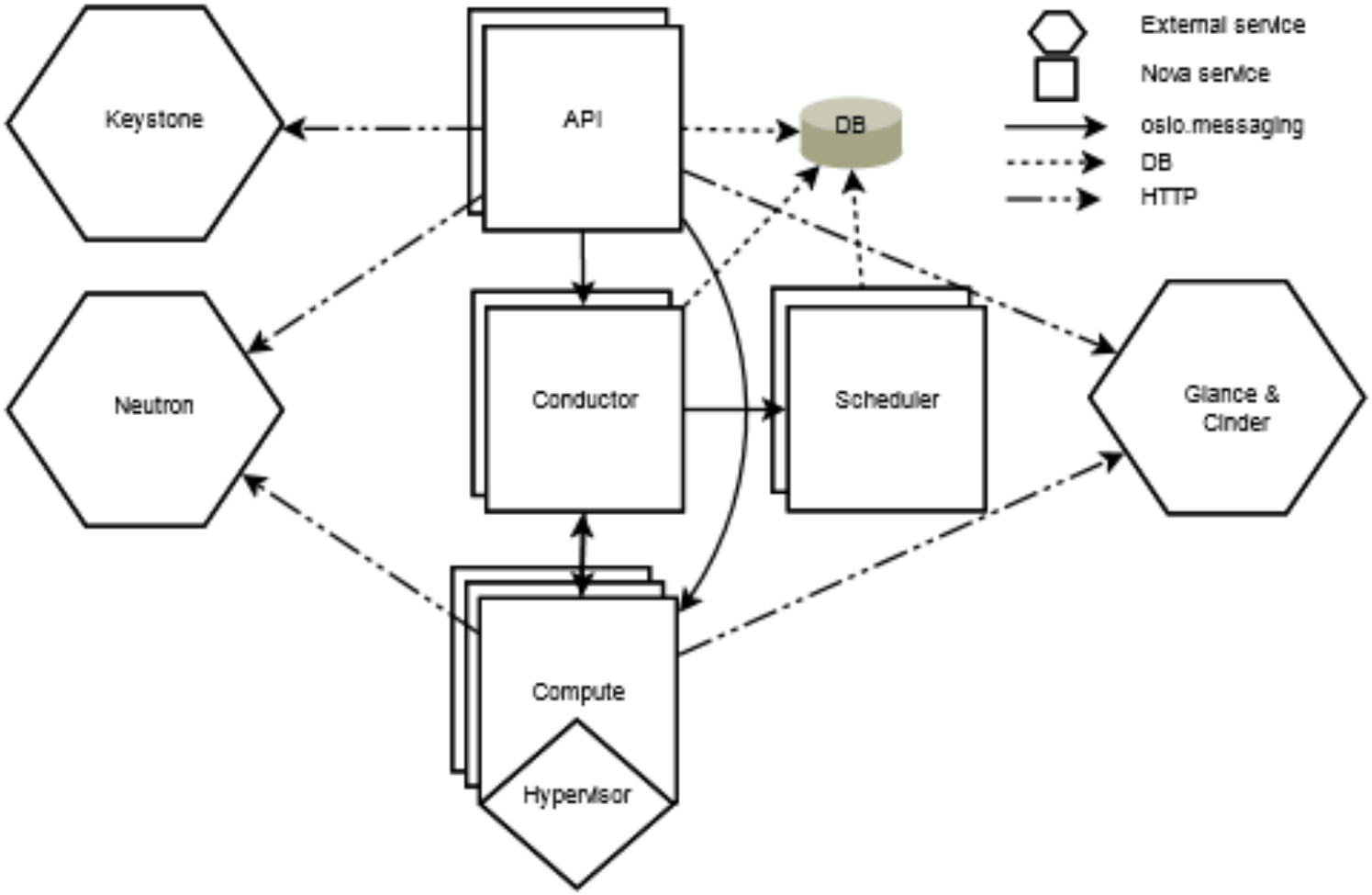
- Controller: centralized service
  - Database: MySQL
  - Message Queue: RabbitMQ
  - Identity: Keystone
  - Networking: Neutron-server/ DHCP agent/ Metadata agent
  - Compute: Nova API/ Conductor/Scheduler
  - Block storage: Cinder Scheduler
  - Etc…
- Compute
  - nova compute, hypervisor, neutron agent
- Block Storage
  - Backend target service, cinder volume etc…



NOVA MAIN SERVICES

- API
  - Component that receives HTTP requests, converts commands and communicates with other components via the **oslo.messaging** queue or HTTP
  - Task orchestration to conductor, short request to compute
- Conductor
  - Task orchestration(task management)
  - Proxy of db access from other node(non-controller node)
- Compute
  - Communicate with hypervisor (via Virt Driver) and VM
- Scheduler
  - Decides which host gets each instance by filtering and weighing

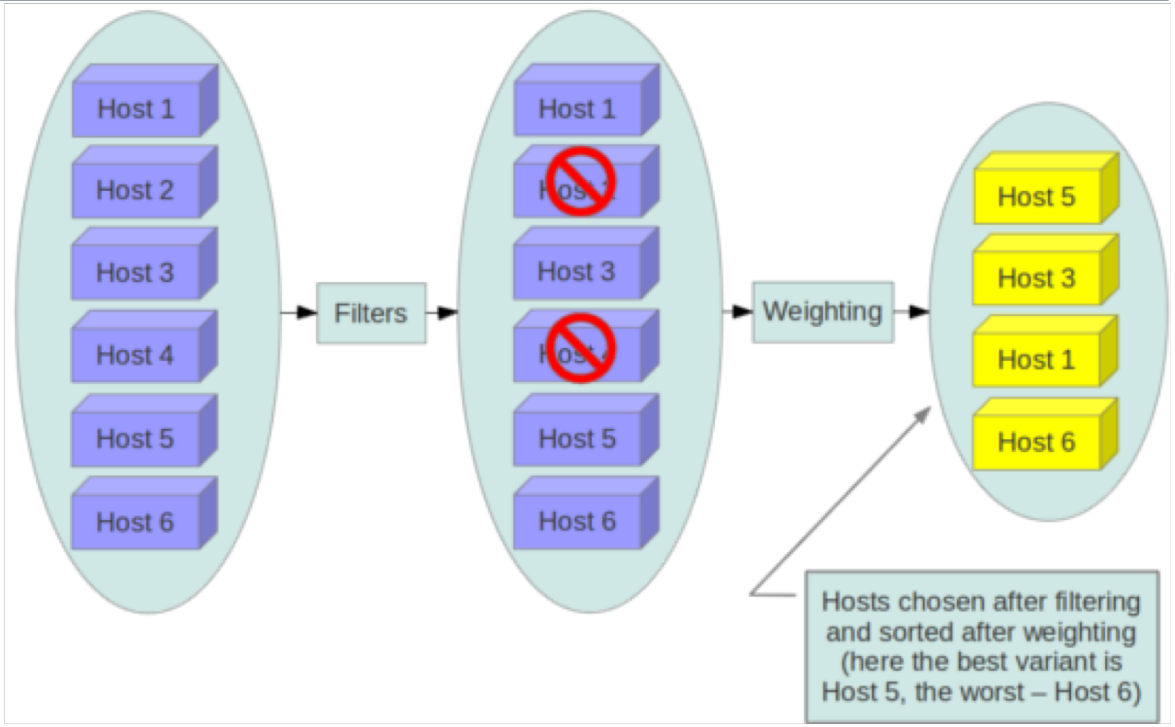
NOVA ARCH.


Keystone
API
DB
External service
Nova service
oslo.messaging
DB
HTTP
Neutron
Conductor
Scheduler
Glance &
Cinder
Compute
Hypervisor

NOVA-SCHEDULER

- FilterScheduler
  - The default one, filtering specific filters

```
[DEFAULT]
scheduler_driver= filter_scheduler
```



COMPUTE: RESOURCE TRACER

- Claim(compute push) nova/compute/resource_tracker.py:instance_claim
  - Before actually building, compute will check resource if enough for this scheduled instace creation?
    - If yes claim it to occupy resources from DB first, then proceeding afterwards:
      - If Instance created, pass
      - Else Instance building failure, it will remove the occupation from DB(via conductor)
    - If not enough, return none directly, no occupation and raise ComputeResourceUavailable exception.
- Periodic Task(db – compute hook)
  - nova/compute/manager.py:class:ComputeManager:update_available_resource
  - It will update resource periodicly

BOOT A VM: NOVA INSTANCE

› Use OSC(OpenStack Client)

```
$ openstack server create --flavor <flavor> --image <image> --nic net-
id=<network_uuid> myFirstVM
```

› Use Novaclient for CLI

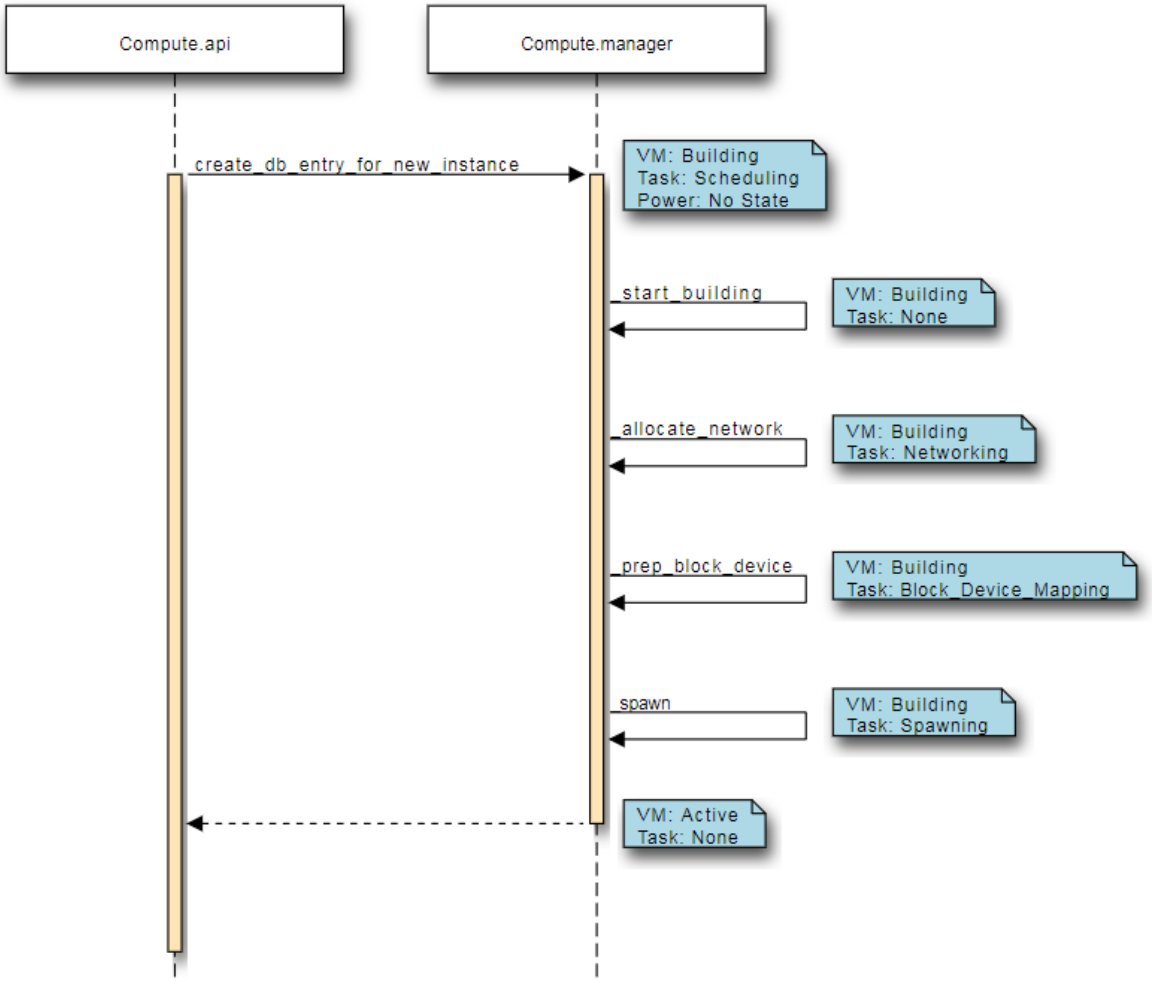
```
$ nova boot --flavor <flavor> --image <image> --nic net-id=<network_uuid>
myFirstVM
```

› RESTful API

```
$ curl –X POST http://controller-fqdn:8774/v2.1/<tenant_id>/servers ...
```

› Horizon

INSTANCE STATE



MYSQL

```
# mysql
MariaDB []> use nova;
MariaDB [nova]> show tables;
+-------------------------------------------+
| Tables_in_nova                            |
+-------------------------------------------+
| agent_builds                              |
| instance_types                            |
...
| instances                                 |
...
| inventories                               |
+-------------------------------------------+
MariaDB [nova]> describe instances;
+--------------------------+----------------------+------+-----+---------+
| Field                    | Type                 | Null | Key | Default |
+--------------------------+----------------------+------+-----+---------+
| created_at               | datetime             | YES  |     | NULL    |
| updated_at               | datetime             | YES  | MUL | NULL    |
| deleted_at               | datetime             | YES  |     | NULL    |
| id                       | int(11)              | NO   | PRI | NULL    |
...
MariaDB [nova]> select created_at from instances where id = ‘<instance_id>’;
```

KVM/QEMU

› virsh
› qemu-img

Look into a kvm/qemu instance via virsh edit <id>

QUESTION: What is tap, how the package goes E/W N/S ?

NETWORKING

- Ethernet/L2/Local Network/broadcast domain(ff:ff…ff)
  - VLAN
- Subnet, ARP
  - The Address Resolution Protocol (ARP) bridges the gap between Ethernet and IP by translating IP addresses into MAC addresses.
  - IP addresses are broken up into two parts: a *network number* and a *host identifier*. Two hosts are on the same *subnet* if they have the same network number. Recall that two hosts can only communicate directly over Ethernet if they are on the same local network. ARP assumes that all machines that are in the same subnet are on the same local network.
- DHCP (client:68 with mac→255.255.255.255, server:67→ mac+ip offer client)
  - Hosts connected to a network use the Dynamic Host Configuration Protocol (DHCP) to dynamically obtain IP addresses. A DHCP server hands out the IP addresses to network hosts, which are the DHCP clients.
- IP/L3
  - The Internet Protocol (IP) specifies how to route packets between hosts that are connected to different local networks. IP relies on special network hosts called routers or gateways. A router is a host that is connected to at least two local networks and can forward IP packets from one local network to another. A router has multiple IP addresses: one for each of the networks it is connected to.

NETWORKING

- Switches
  - Switches are Multi-Input Multi-Output (MIMO) devices that enable packets to travel from one node to another. Switches connect hosts that belong to the same layer-2 network. Switches enable forwarding of the packet received on one port (input) to another port (output) so that they reach the desired destination node. Switches operate at layer-2 in the networking model. They forward the traffic based on the destination Ethernet address in the packet header.
- Routers
  - Routers are special devices that enable packets to travel from one layer-3 network to another. Routers enable communication between two nodes on different layer-3 networks that are not directly connected to each other. Routers operate at layer-3 in the networking model. They route the traffic based on the destination IP address in the packet header.
- Firewalls
  - Firewalls are used to regulate traffic to and from a host or a network. A firewall can be either a specialized device connecting two networks or a software-based filtering mechanism implemented on an operating system. Firewalls are used to restrict traffic to a host based on the rules defined on the host. They can filter packets based on several criteria such as source IP address, destination IP address, port numbers, connection state, and so on. It is primarily used to protect the hosts from unauthorized access and malicious attacks. Linux-based operating systems implement firewalls through iptables.

NETOWORKING SEGMENT

- network namespace (network node)
  - https://docs.openstack.org/neutron/latest/admin/intro-network-namespaces.html
- Overlay protocol
  - VLAN (L2)
  - VXLAN,GRE

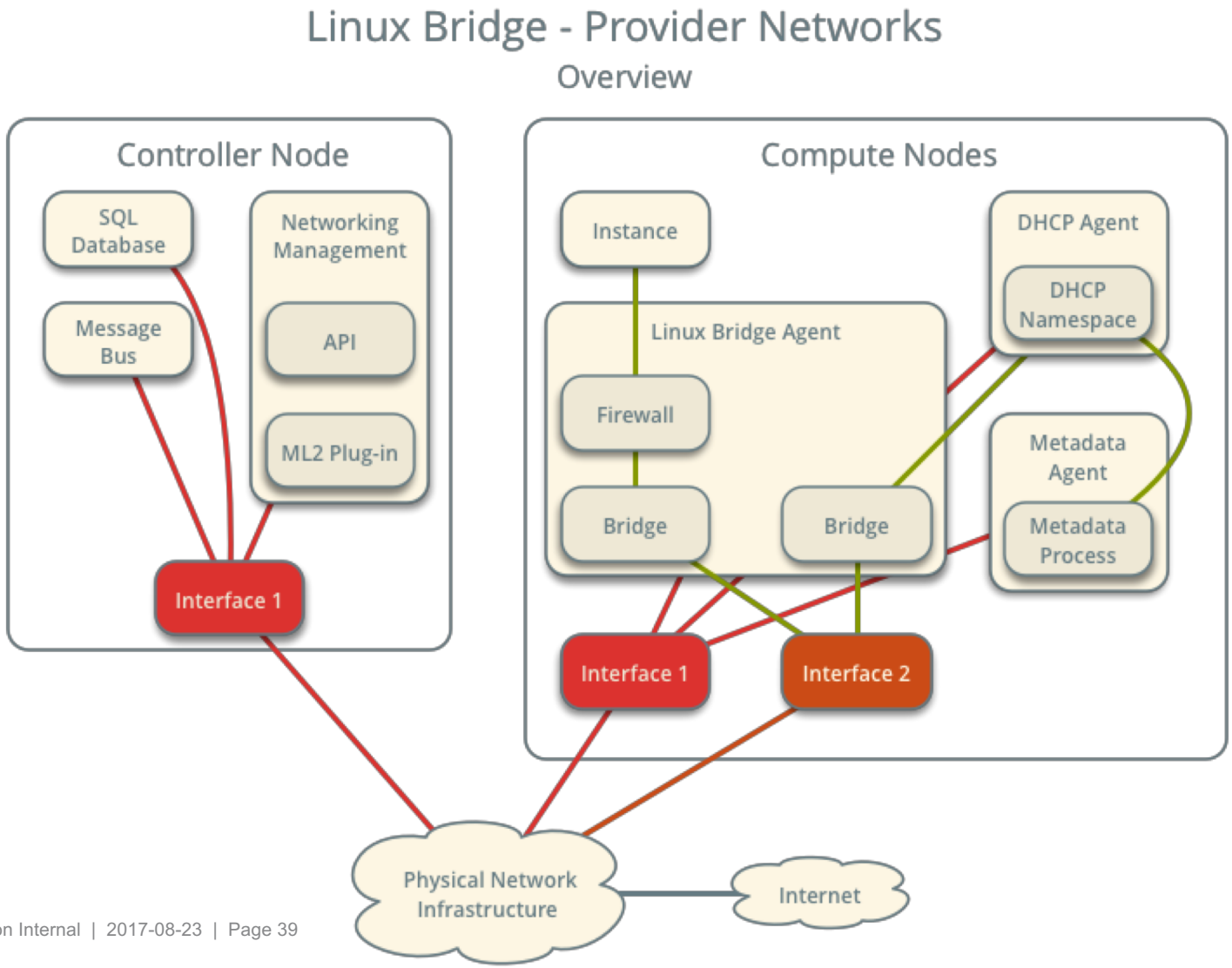
TYPES OF NETWORK

- Provider networks
- Self-service networks

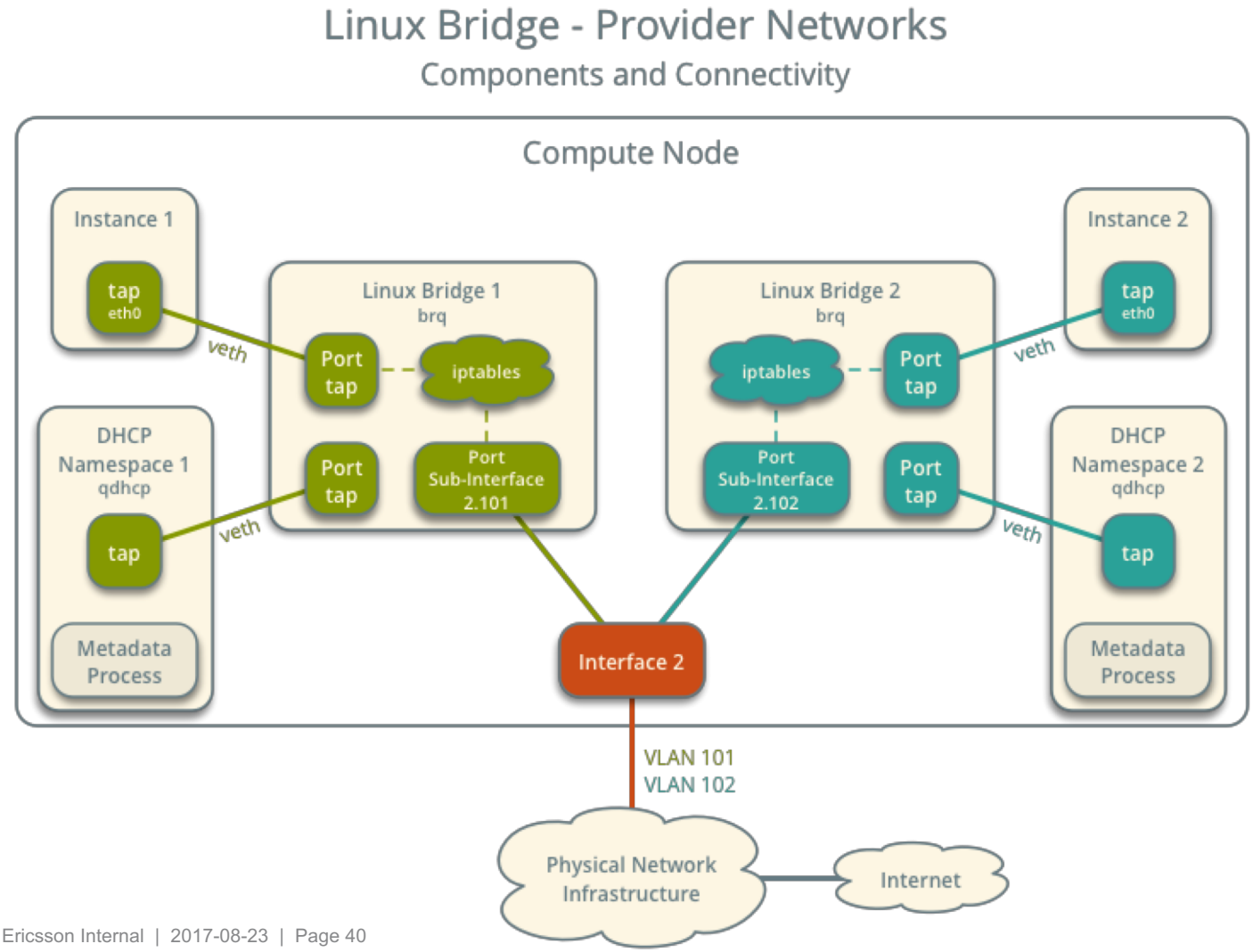
NEUTRON CONCEPTS

- Subnets
  - A block of IP addresses and associated configuration state. This is also known as the native IPAM (IP Address Management) provided by the networking service for both project and provider networks. Subnets are used to allocate IP addresses when new ports are created on a network.
- Ports
  - A port is a connection point for attaching a single device, such as the NIC of a virtual server, to a virtual network. The port also describes the associated network configuration, such as the MAC and IP addresses to be used on that port.
- Routers
  - Routers provide virtual layer-3 services such as routing and NAT between self-service and provider networks or among self-service networks belonging to a project. The Networking service uses a layer-3 agent to manage routers via namespaces.

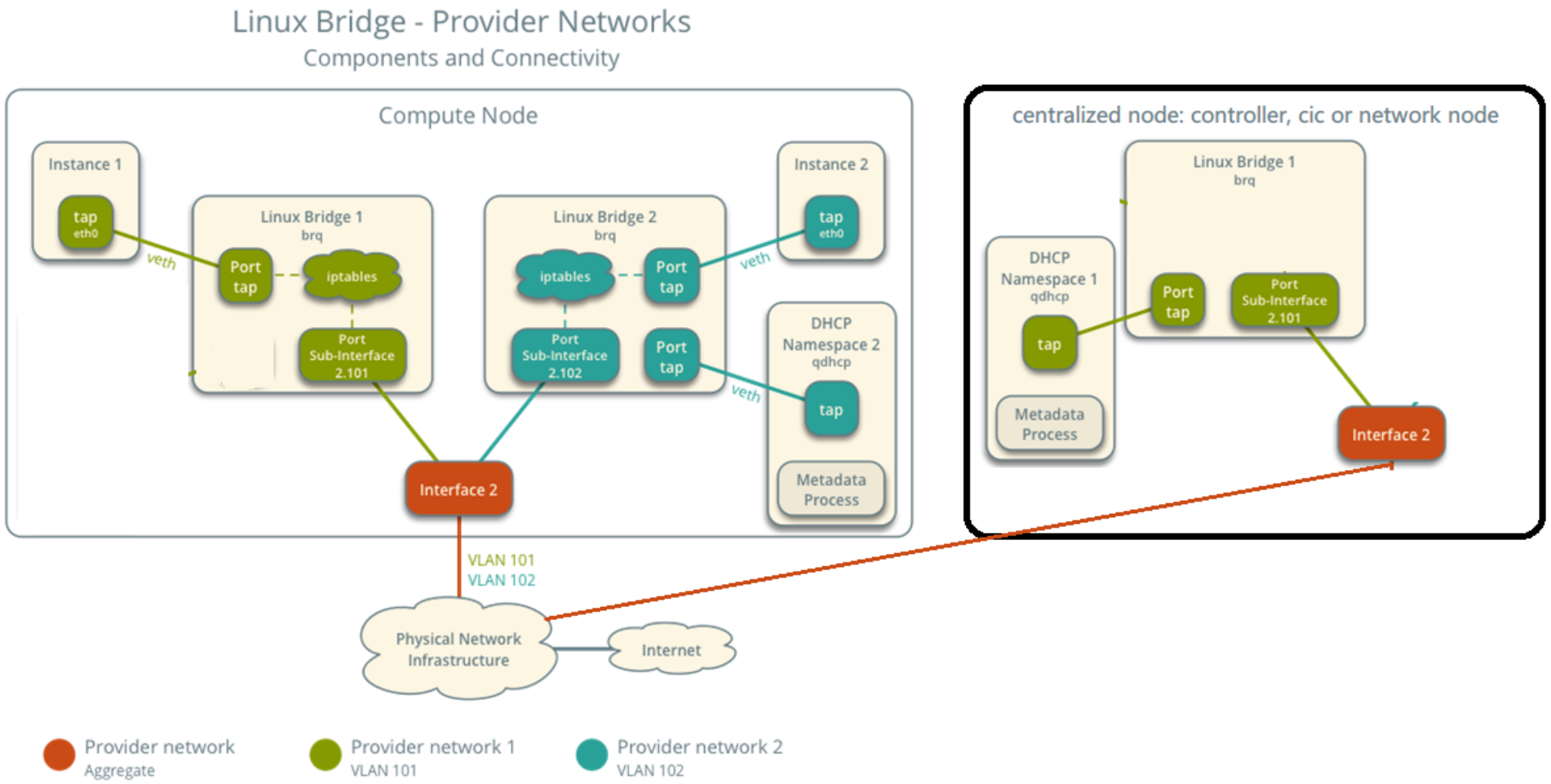
LINUX BRIDGE BASED NETWORK OVERVIEW



LINUX BRIDGE BASED NETWORK IN DETAILED

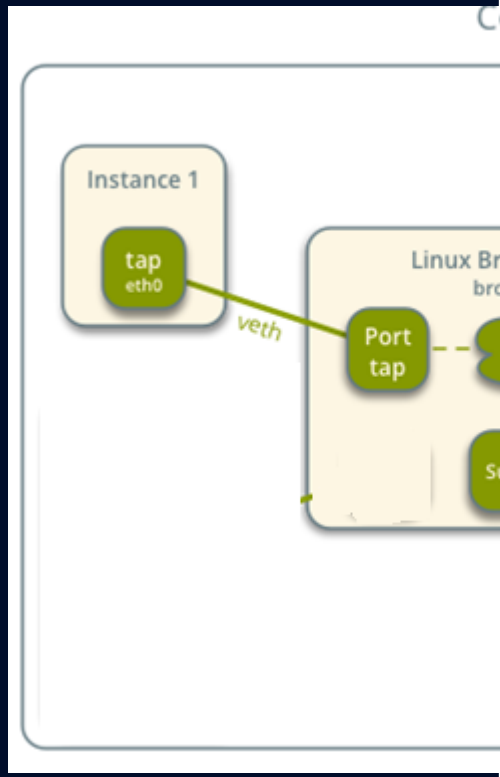


DHCP AGENT IN DIFFERENT NODE FROM INSTANCE



INSTANCE(VM) NETWORK

```
// Console log piece
### ifconfig -a
eth0      Link encap:Ethernet  HWaddr FA:16:3E:66:84:CC
          inet addr:172.16.0.105  Bcast:172.16.0.255  Mask:255.255.255.0
          inet6 addr: fe80::f816:3eff:fe66:84cc/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:69 errors:0 dropped:0 overruns:0 frame:0
          TX packets:102 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:7404 (7.2 KiB)  TX bytes:10322 (10.0 KiB)
...
### route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0         172.16.0.1      0.0.0.0         UG    0      0        0 eth0
169.254.169.254 172.16.0.100    255.255.255.255 UGH   0      0        0 eth0
172.16.0.0      0.0.0.0         255.255.255.0   U     0      0        0 eth0
// Console log piece end
root@compute:~# virsh dumpxml 1 | grep -i "FA:16:3E:66:84:CC" -A7 -B1
    <interface type='bridge'>
      <mac address='fa:16:3e:66:84:cc'/>
      <source bridge='brq152a4a85-dc'/>
      <target dev='tap584e904e-a1'/>
      <model type='virtio'/>
      <driver name='qemu'/>
      <alias name='net0'/>
      <address type='pci' domain='0x0000' bus='0x00' slot='0x03' function='0x0'/>
    </interface>
```

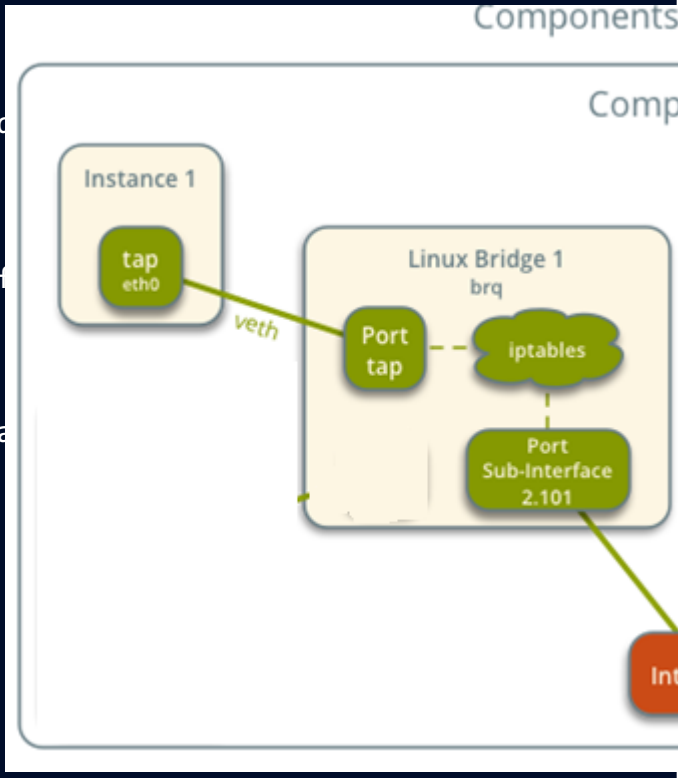


COMPUTE(VM HOST) NET

```
root@compute:~# ip -d link show enp0s8.101
10: enp0s8.101@enp0s8: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue master brq152a4a85-dc
state UP mode DEFAULT group default qlen 1000
    link/ether 08:00:27:95:2e:64 brd ff:ff:ff:ff:ff:ff promiscuity 1
    vlan protocol 802.1Q id 101 <REORDER_HDR>
    bridge_slave state forwarding priority 32 cost 4 hairpin off guard
learning on flood on addrgenmode eui64

root@compute:~# ip -d link show tap584e904e-a1
9: tap584e904e-a1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pf
UNKNOWN mode DEFAULT group default qlen 1000
    link/ether fe:16:3e:66:84:cc brd ff:ff:ff:ff:ff:ff promiscuity 1
    tun
    bridge_slave state forwarding priority 32 cost 100 hairpin off gua
learning on flood on addrgenmode eui64

root@compute:~# ethtool -i tap584e904e-a1
driver: tun
version: 1.6
firmware-version:
expansion-rom-version:
bus-info: tap
supports-statistics: no
supports-test: no
supports-eeprom-access: no
supports-register-dump: no
supports-priv-flags: no
```



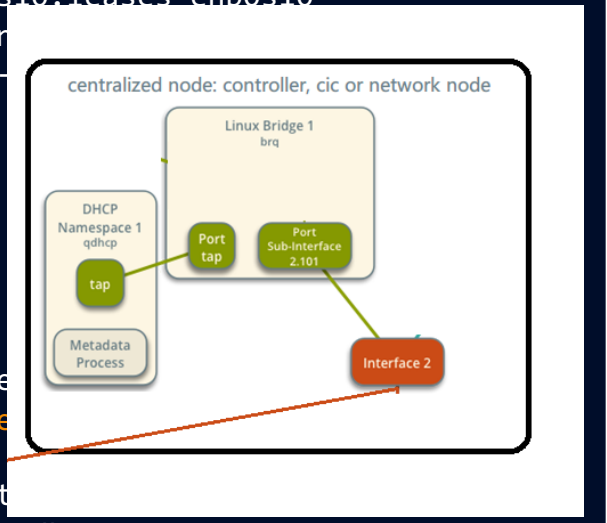
IPTABLE (SECURITY GROUP :FIREWALL)

```
root@compute:~# iptables -S | grep 584e904e-a
-N neutron-linuxbri-i584e904e-a
-N neutron-linuxbri-o584e904e-a
-N neutron-linuxbri-s584e904e-a
-A neutron-linuxbri-FORWARD -m physdev --physdev-out tap584e904e-a1 --physdev-is-bridged -m comment --comment "Direct traffic from the VM interface to the
security group chain." -j neutron-linuxbri-sg-chain
-A neutron-linuxbri-FORWARD -m physdev --physdev-in tap584e904e-a1 --physdev-is-bridged -m comment --comment "Direct traffic from the VM interface to the
security group chain." -j neutron-linuxbri-sg-chain
-A neutron-linuxbri-INPUT -m physdev --physdev-in tap584e904e-a1 --physdev-is-bridged -m comment --comment "Direct incoming traffic from VM to the security
group chain." -j neutron-linuxbri-o584e904e-a

-A neutron-linuxbri-i584e904e-a -m state --state RELATED,ESTABLISHED -m comment --comment "Direct packets associated with a known session to the RETURN
chain." -j RETURN
-A neutron-linuxbri-i584e904e-a -d 172.16.0.105/32 -p udp -m udp --sport 67 --dport 68 -j RETURN
-A neutron-linuxbri-i584e904e-a -d 255.255.255.255/32 -p udp -m udp --sport 67 --dport 68 -j RETURN
-A neutron-linuxbri-i584e904e-a -m set --match-set NIPv4fa58711d-9bf5-4d94-9fd6- src -j RETURN
-A neutron-linuxbri-i584e904e-a -p tcp -m tcp --dport 22 -j RETURN
-A neutron-linuxbri-i584e904e-a -p icmp -j RETURN
-A neutron-linuxbri-i584e904e-a -m state --state INVALID -m comment --comment "Drop packets that appear related to an existing connection (e.g. TCP ACK/FIN)
but do not have an entry in conntrack." -j DROP
-A neutron-linuxbri-i584e904e-a -m comment --comment "Send unmatched traffic to the fallback chain." -j neutron-linuxbri-sg-fallback
-A neutron-linuxbri-o584e904e-a -s 0.0.0.0/32 -d 255.255.255.255/32 -p udp -m udp --sport 68 --dport 67 -m comment --comment "Allow DHCP client traffic." -j
RETURN
-A neutron-linuxbri-o584e904e-a -j neutron-linuxbri-s584e904e-a
-A neutron-linuxbri-o584e904e-a -p udp -m udp --sport 68 --dport 67 -m comment --comment "Allow DHCP client traffic." -j RETURN
-A neutron-linuxbri-o584e904e-a -p udp -m udp --sport 67 --dport 68 -m comment --comment "Prevent DHCP Spoofing by VM." -j DROP
-A neutron-linuxbri-o584e904e-a -m state --state RELATED,ESTABLISHED -m comment --comment "Direct packets associated with a known session to the RETURN
chain." -j RETURN
-A neutron-linuxbri-o584e904e-a -j RETURN
-A neutron-linuxbri-o584e904e-a -m state --state INVALID -m comment --comment "Drop packets that appear related to an existing connection (e.g. TCP ACK/FIN)
but do not have an entry in conntrack." -j DROP
-A neutron-linuxbri-o584e904e-a -m comment --comment "Send unmatched traffic to the fallback chain." -j neutron-linuxbri-sg-fallback
-A neutron-linuxbri-s584e904e-a -s 172.16.0.105/32 -m mac --mac-source FA:16:3E:66:84:CC -m comment --comment "Allow traffic from defined IP/MAC pairs." -j
RETURN
-A neutron-linuxbri-s584e904e-a -m comment --comment "Drop traffic without an IP/MAC allow rule." -j DROP
-A neutron-linuxbri-sg-chain -m physdev --physdev-out tap584e904e-a1 --physdev-is-bridged -m comment --comment "Jump to the VM specific chain." -j neutron-
linuxbri-i584e904e-a
-A neutron-linuxbri-sg-chain -m physdev --physdev-in tap584e904e-a1 --physdev-is-bridged -m comment --comment "Jump to the VM specific chain." -j neutron-
linuxbri-o584e904e-a
```

CONTROLLER (DHCP PROCESS)

```
root@controller:~# ps -ef | grep dhcp
root      1322     1  0 12:25 ?        00:00:00 /sbin/dhclient -1 -v -pf /run/dhclient.enp0s10.pid -lf
/var/lib/dhcp/dhclient.enp0s10.leases -I -df /var/lib/dhcp/dhclient6.enp0s10.leases enp0s10
neutron   2707     1  0 12:25 ?        00:02:47 /usr/bin/python /usr/bin/r
file=/etc/neutron/neutron.conf --config-file=/etc/neutron/dhcp_agent.ini -
file=/var/log/neutron/neutron-dhcp-agent.log
libvirt+  2929     1  0 12:25 ?        00:00:00 /usr/sbin/dnsmasq --conf-
file=/var/lib/libvirt/dnsmasq/default.conf --leasefile-ro --dhcp-
script=/usr/lib/libvirt/libvirt_leaseshelper
root      2930  2929  0 12:25 ?        00:00:00 /usr/sbin/dnsmasq --conf-
file=/var/lib/libvirt/dnsmasq/default.conf --leasefile-ro --dhcp-
script=/usr/lib/libvirt/libvirt_leaseshelper
nobody    3690     1  0 12:29 ?        00:00:00 dnsmasq --no-hosts --no-re
interface=lo --pid-file=/var/lib/neutron/dhcp/152a4a85-dc52-4c62-9bbd-742e
hostsfile=/var/lib/neutron/dhcp/152a4a85-dc52-4c62-9bbd-742eb4f7b8fa/host
hosts=/var/lib/neutron/dhcp/152a4a85-dc52-4c62-9bbd-742eb4f7b8fa/addn_host
optsfile=/var/lib/neutron/dhcp/152a4a85-dc52-4c62-9bbd-742eb4f7b8fa/opts --dhcp-
leasefile=/var/lib/neutron/dhcp/152a4a85-dc52-4c62-9bbd-742eb4f7b8fa/leases --dhcp-match=set:ipxe,175 --
bind-interfaces --interface=ns-edebcfe1-74 --dhcp-range=set:tag0,172.16.0.0,static,86400s --dhcp-option-
force=option:mtu,1500 --dhcp-lease-max=256 --conf-file= --domain=openstacklocal
root      8499  3059  0 18:35 pts/0    00:00:00 grep --color=auto dhcp
```

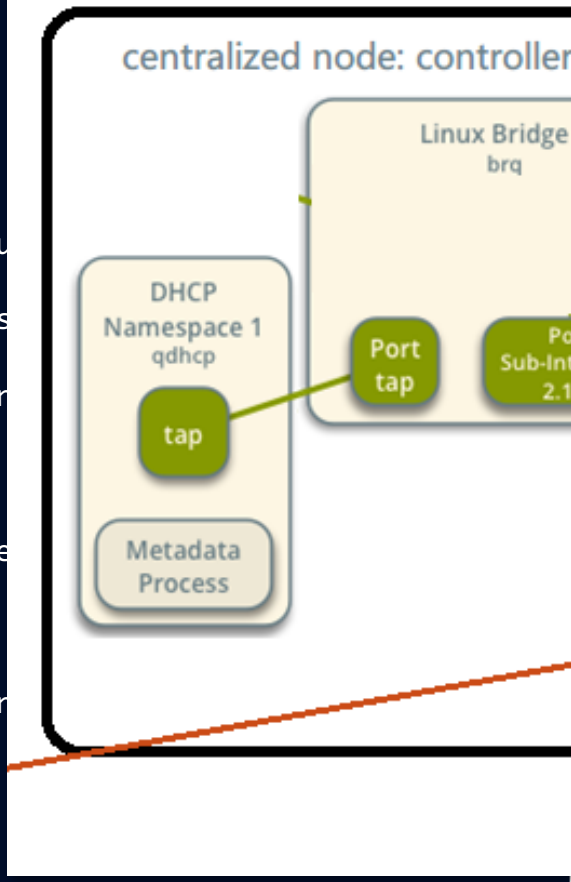


CONTROLLER (NETWORK, PROJECT NET: VLAN)

```
root@controller:~# brctl show
bridge name     bridge id               STP enabled     interfaces
brq152a4a85-dc          8000.080027ad76b4       no              enp0s8.101
                                                        tapedebcfe1-74
virbr0          8000.525400c2b2e9       yes             virbr0-nic

root@controller:~# ip -d link show tapedebcfe1-74
8: tapedebcfe1-74@if2: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqu
state UP mode DEFAULT group default qlen 1000
    link/ether 7a:3e:4f:ae:7e:23 brd ff:ff:ff:ff:ff:ff link-netnsid 0 promis
    veth
    bridge_slave state forwarding priority 32 cost 2 hairpin off guard off r
learning on flood on addrgenmode eui64

root@controller:~# ip -d link show enp0s8.101
9: enp0s8.101@enp0s8: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noque
UP mode DEFAULT group default qlen 1000
    link/ether 08:00:27:ad:76:b4 brd ff:ff:ff:ff:ff:ff promiscuity 1
    vlan protocol 802.1Q id 101 <REORDER_HDR>
    bridge_slave state forwarding priority 32 cost 4 hairpin off guard off r
learning on flood on addrgenmode eui64
```

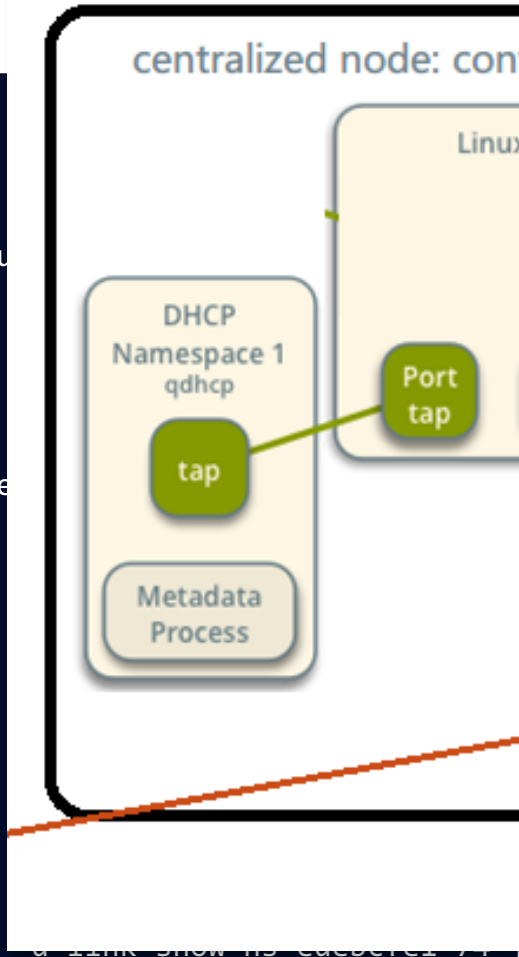


CONTROLLER (DHCP SERVER)

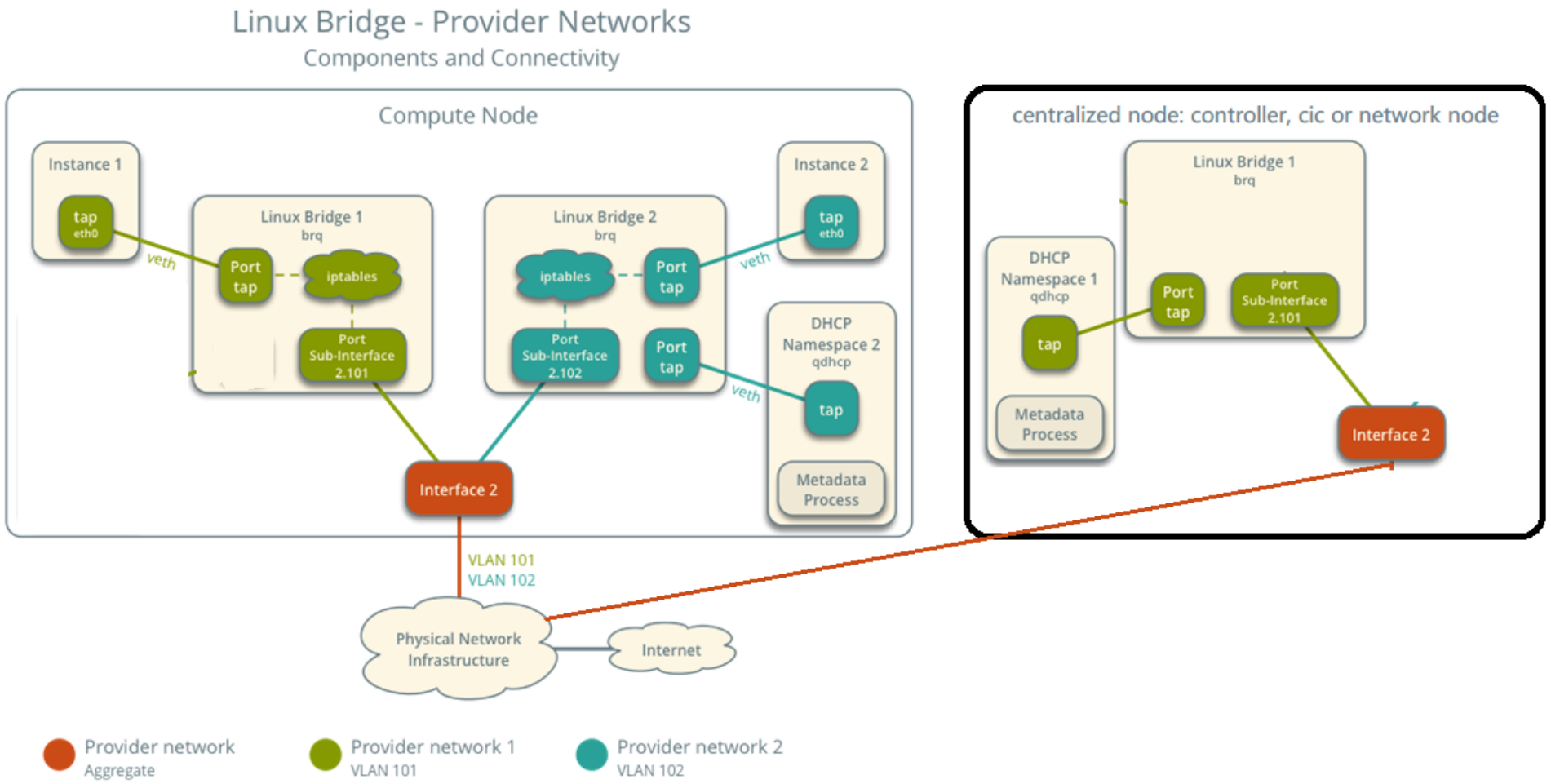
```
root@controller:~# ip netns
qdhcp-152a4a85-dc52-4c62-9bbd-742eb4f7b8fa (id: 0)

root@controller:~# ip netns exec qdhcp-152a4a85-dc52-4c62-9bbd-742eb4f7b8fa ip
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group defau
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: ns-edebcfe1-74@if8: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue
qlen 1000
    link/ether fa:16:3e:56:9f:61 brd ff:ff:ff:ff:ff:ff link-netnsid 0
    inet 169.254.169.254/16 brd 169.254.255.255 scope global ns-edebcfe1-74
       valid_lft forever preferred_lft forever
    inet 172.16.0.100/24 brd 172.16.0.255 scope global ns-edebcfe1-74
       valid_lft forever preferred_lft forever
    inet6 fe80::f816:3eff:fe56:9f61/64 scope link
       valid_lft forever preferred_lft forever

root@controller:~# ip -d link show ns-edebcfe1-74
Device "ns-edebcfe1-74" does not exist.
root@controller:~# ip netns exec qdhcp-152a4a85-dc52-4c62-9bbd-742eb4f7b8fa ip
2: ns-edebcfe1-74@if8: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP mode DEFAULT
group default qlen 1000
    link/ether fa:16:3e:56:9f:61 brd ff:ff:ff:ff:ff:ff link-netnsid 0 promiscuity 0
    veth addrgenmode eui64
```



DHCP AGENT IN DIFFERENT NODE FROM INSTANCE



CONNECT TO PROJECT NETWORK VIA DHCP NAMESPACE

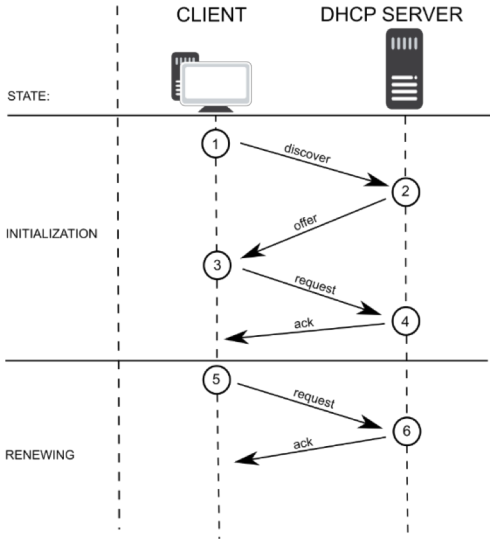
```
root@controller:~# ip netns exec qdhcp-152a4a85-dc52-4c62-9bbd-742eb4f7b8fa ping -c 1 172.16.0.105
PING 172.16.0.105 (172.16.0.105) 56(84) bytes of data.
64 bytes from 172.16.0.105: icmp_seq=1 ttl=64 time=5.89 ms

--- 172.16.0.105 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 5.893/5.893/5.893/0.000 ms

root@controller:~# ip netns exec qdhcp-152a4a85-dc52-4c62-9bbd-742eb4f7b8fa ssh cirros@172.16.0.105
cirros@172.16.0.105's password:
$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 16436 qdisc noqueue
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast qlen 1000
    link/ether fa:16:3e:66:84:cc brd ff:ff:ff:ff:ff:ff
    inet 172.16.0.105/24 brd 172.16.0.255 scope global eth0
    inet6 fe80::f816:3eff:fe66:84cc/64 scope link
       valid_lft forever preferred_lft forever
```

VM DHCP TROUBLESHOOTING

- dnsmasq log:
  - /var/log/syslog (CEE /var/log/damon.log)
- tcpdump port 67 & 68
  - tcpdump -i <brq2a33434f-ba> -vv port 67 or port 68 -e –n
- Use network namespace to tcpdump dhcp server
  - ip netns exec <ns> tcpdump -i <nstap> -vv port 67 or port 68 -e –n
- Flow



OVERLAY (TUNNEL) PROTOCOLS

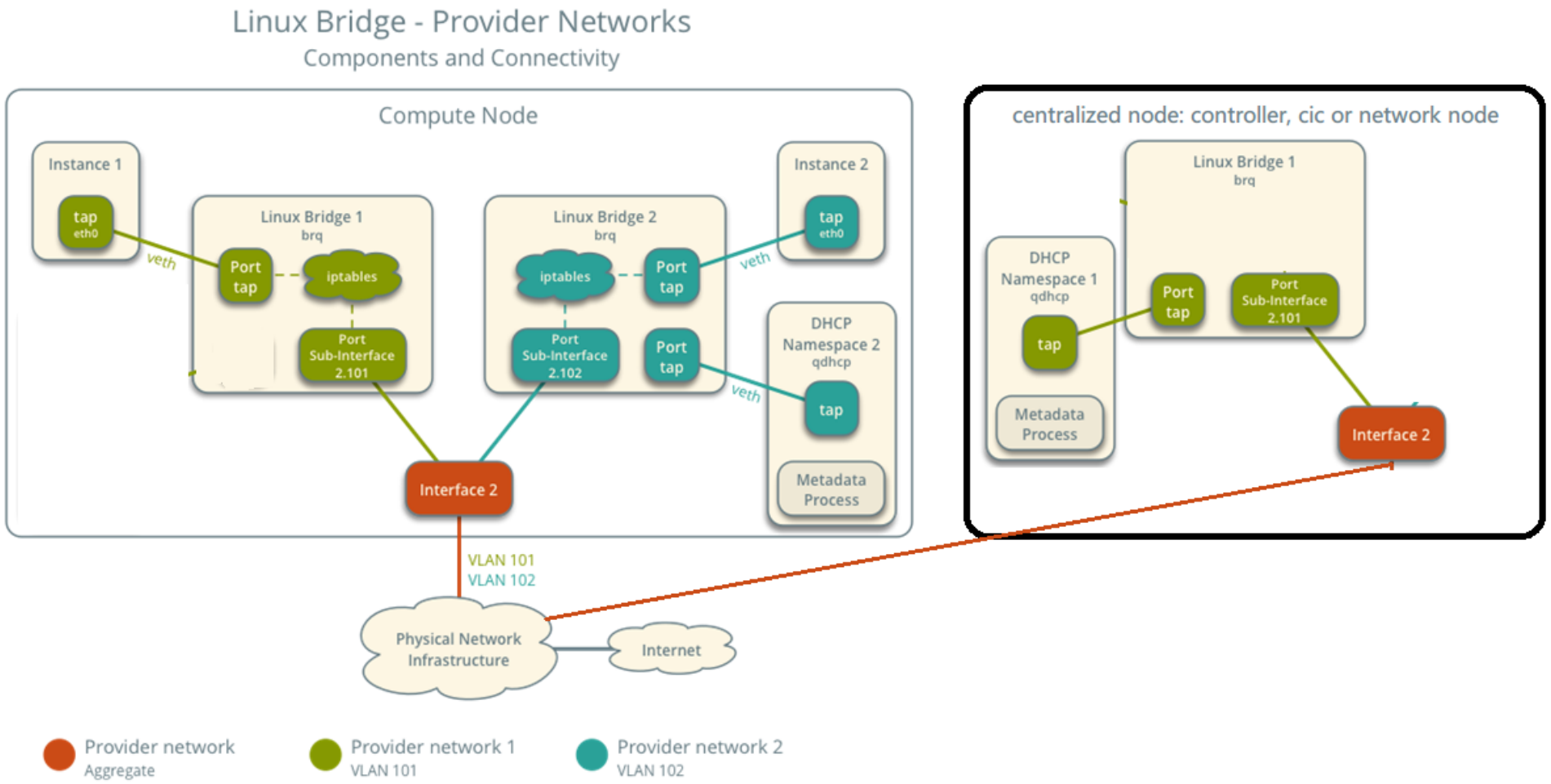
Tunneling is a mechanism that makes transfer of payloads feasible over an incompatible delivery network. It allows the network user to gain access to denied or insecure networks. Data encryption may be employed to transport the payload, ensuring that the encapsulated user network data appears as public even though it is private and can easily pass the conflicting network.

- Generic routing encapsulation (GRE) (ovs)
  - Generic routing encapsulation (GRE) is a protocol that runs over IP and is employed when delivery and payload protocols are compatible but payload addresses are incompatible. For instance, a payload might think it is running on a datalink layer but it is actually running over a transport layer using datagram protocol over IP. GRE creates a private point-to-point connection and works by encapsulating a payload. GRE is a foundation protocol for other tunnel protocols but the GRE tunnels provide only weak authentication.
- Virtual extensible local area network (VXLAN) (ovs, linux bridge)
  - The purpose of VXLAN is to provide scalable network isolation. VXLAN is a Layer 2 overlay scheme on a Layer 3 network. It allows an overlay layer-2 network to spread across multiple underlay layer-3 network domains. Each overlay is termed a VXLAN segment. Only VMs within the same VXLAN segment can communicate.

NEUTRON SERVICE

- Server
- Plugin
- Agent
- Message queue
- DB

DHCP AGENT IN DIFFERENT NODE FROM INSTANCE



NEUTRON

- Openvswitch DPDK
  - https://github.com/openvswitch/ovs/blob/master/Documentation/howto/dpdk.rst
- SR-IOV
  - https://docs.openstack.org/mitaka/networking-guide/config-sriov.html

BLOCK STORAGE, CINDER

The OpenStack Block Storage service (cinder) adds persistent storage to a virtual machine. Block Storage provides an infrastructure for managing volumes, and interacts with OpenStack Compute to provide volumes for instances. The service also enables management of volume snapshots, and volume types.
The Block Storage service consists of the following components:

- cinder-api
  - Accepts API requests, and routes them to the cinder-volume for action.
- cinder-volume
  - Interacts directly with the Block Storage service, and processes such as the cinder-scheduler. It also interacts with these processes through a message queue. The cinder-volume service responds to read and write requests sent to the Block Storage service to maintain state. It can interact with a variety of storage providers through a driver architecture.
- cinder-scheduler daemon
  - Selects the optimal storage provider node on which to create the volume. A similar component to the nova-scheduler.
- cinder-backup daemon
  - The cinder-backup service provides backing up volumes of any type to a backup storage provider. Like the cinder-volume service, it can interact with a variety of storage providers through a driver architecture.
- Messaging queue
  - Routes information between the Block Storage processes.

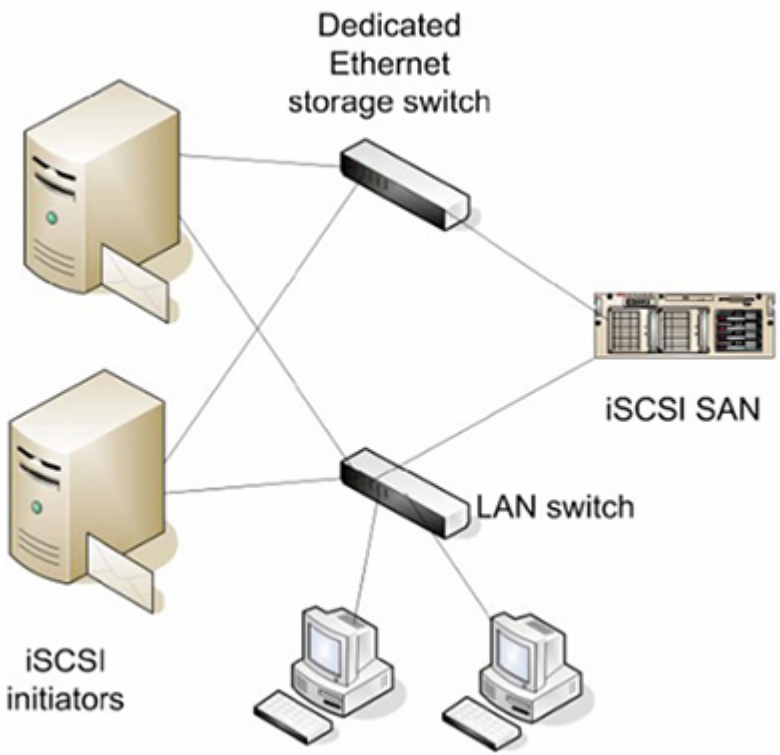
ABOUT DISK DEVICES IN VIRSH DUMPXML 1

```
root@compute:~# qemu-img info /var/lib/nova/instances/e73c64ac-3af3-47fd-abfe-e138e40f0a40/disk
image: /var/lib/nova/instances/e73c64ac-3af3-47fd-abfe-e138e40f0a40/disk
file format: qcow2
virtual size: 1.0G (1073741824 bytes)
disk size: 2.4M
cluster_size: 65536
backing file: /var/lib/nova/instances/_base/cb8c547f24446c1fb2a29358f5adea4874f0931d
Format specific information:
    compat: 1.1
    lazy refcounts: false
    refcount bits: 16
    corrupt: false

root@compute:~# qemu-img info /var/lib/nova/instances/_base/cb8c547f24446c1fb2a29358f5adea4874f0931d
image: /var/lib/nova/instances/_base/cb8c547f24446c1fb2a29358f5adea4874f0931d
file format: raw
virtual size: 39M (41126400 bytes)
disk size: 18M
root@compute:~# qemu-img info /dev/disk/by-path/ip-192.168.199.30:3260-iscsi-iqn.2010-
10.org.openstack:volume-81ffed40-ed71-495d-bfa9-8fb8c72cf222-lun-1
image: /dev/disk/by-path/ip-192.168.199.30:3260-iscsi-iqn.2010-10.org.openstack:volume-81ffed40-ed71-495d-
bfa9-8fb8c72cf222-lun-1
file format: raw
virtual size: 1.0G (1073741824 bytes)
disk size: 0
```

ISCSI

- iSCSI target (SAN)
  - iqn.yyyy-mm.<reversed domain name>:identifier
- iSCSI initiator (Consumer)
  - /dev/sd<x>
- LUN
  - Logic unit number

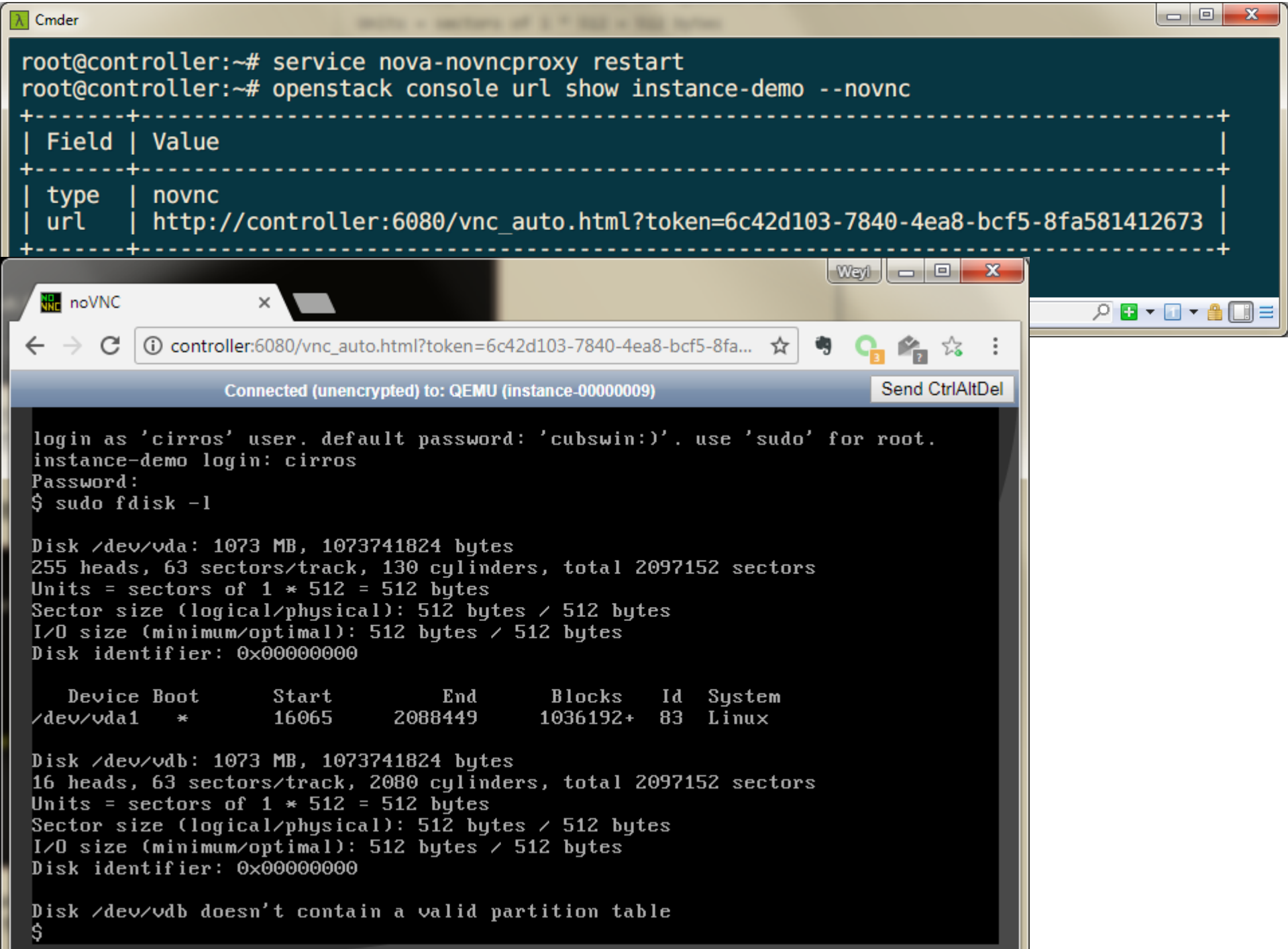


CHECK A VOLUME

```
root@compute:~# qemu-img info /var/lib/nova/instances/e73c64ac-3af3-47fd-abfe-e138e40f0a40/disk
image: /var/lib/nova/instances/e73c64ac-3af3-47fd-abfe-e138e40f0a40/disk
file format: qcow2
virtual size: 1.0G (1073741824 bytes)
disk size: 2.4M
cluster_size: 65536
backing file: /var/lib/nova/instances/_base/cb8c547f24446c1fb2a29358f5adea4874f0931d
Format specific information:
    compat: 1.1
    lazy refcounts: false
    refcount bits: 16
    corrupt: false

root@compute:~# qemu-img info /var/lib/nova/instances/_base/cb8c547f24446c1fb2a29358f5adea4874f0931d
image: /var/lib/nova/instances/_base/cb8c547f24446c1fb2a29358f5adea4874f0931d
file format: raw
virtual size: 39M (41126400 bytes)
disk size: 18M
root@compute:~# qemu-img info /dev/disk/by-path/ip-192.168.199.30:3260-iscsi-iqn.2010-
10.org.openstack:volume-81ffed40-ed71-495d-bfa9-8fb8c72cf222-lun-1
image: /dev/disk/by-path/ip-192.168.199.30:3260-iscsi-iqn.2010-10.org.openstack:volume-81ffed40-ed71-495d-
bfa9-8fb8c72cf222-lun-1
file format: raw
virtual size: 1.0G (1073741824 bytes)
disk size: 0
```

```
root@storage:~# tgtadm --lld iscsi --op show --mode target
Target 1: iqn.2010-10.org.openstack:volume-81ffed40-ed71-495d-bfa9-8fb8c72cf222
    System information:
        Driver: iscsi
        State: ready
    I_T nexus information:
        I_T nexus: 1
            Initiator: iqn.1993-08.org.debian:01:e7b693dedcab alias: compute
            Connection: 0
                IP Address: 192.168.199.20
    LUN information:
        LUN: 0
            Type: controller
            SCSI ID: IET     00010000
            SCSI SN: beaf10
            Size: 0 MB, Block size: 1
            Online: Yes
            Removable media: No
            Prevent removal: No
            Readonly: No
            SWP: No
            Thin-provisioning: No
            Backing store type: null
            Backing store path: None
            Backing store flags:
        LUN: 1
            Type: disk
            SCSI ID: IET     00010001
            SCSI SN: beaf11
            Size: 1074 MB, Block size: 512
            Online: Yes
            Removable media: No
            Prevent removal: No
            Readonly: No
            SWP: No
            Thin-provisioning: No
            Backing store type: rdwr
            Backing store path: /dev/cinder-volumes/volume-81ffed40-ed71-495d-bfa9-8fb8c72cf222
            Backing store flags:
    Account information:
        7jTnhhxXsVM4BwqxG979
    ACL information:
        ALL
```


Cmder
root@controller:~# service nova-novncproxy restart
root@controller:~# openstack console url show instance-demo --novnc
+-------+---+
| Field | Value |
+-------+---+
| type | novnc |
| url | http://controller:6080/vnc_auto.html?token=6c42d103-7840-4ea8-bcf5-8fa581412673 |
+-------+---+
Wey
noVNC
controller:6080/vnc_auto.html?token=6c42d103-7840-4ea8-bcf5-8fa...
Connected (unencrypted) to: QEMU (instance-00000009)
Send CtrlAltDel
login as 'cirros' user. default password: 'cubswin:)'. use 'sudo' for root.
instance-demo login: cirros
Password:
$ sudo fdisk -l
Disk /dev/vda: 1073 MB, 1073741824 bytes
255 heads, 63 sectors/track, 130 cylinders, total 2097152 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk identifier: 0x00000000
Device Boot Start End Blocks Id System
/dev/vda1 * 16065 2088449 1036192+ 83 Linux
Disk /dev/vdb: 1073 MB, 1073741824 bytes
16 heads, 63 sectors/track, 2080 cylinders, total 2097152 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk identifier: 0x00000000
Disk /dev/vdb doesn't contain a valid partition table
$

CONFIGURATIONS

```
/etc/cinder/cinder.conf
[lvm]
#
...
volume_driver =
cinder.volume.drivers.lvm.LVMVolumeDriver
volume_group = cinder-volumes
iscsi_protocol = iscsi
iscsi_helper = tgtadm

[DEFAULT]
#
...
enabled_backends = lvm
```

ORCHESTRATION HEAT

- Heat's purpose and vision
  - Heat provides a template based orchestration for describing a cloud application by executing appropriate OpenStack API calls to generate running cloud applications.
  - A Heat template describes the infrastructure for a cloud application in text files which are readable and writable by humans, and can be managed by version control tools.
  - Templates specify the relationships between resources (e.g. this volume is connected to this server). This enables Heat to call out to the OpenStack APIs to create all of your infrastructure in the correct order to completely launch your application.
  - The software integrates other components of OpenStack. The templates allow creation of most OpenStack resource types (such as instances, floating ips, volumes, security groups, users, etc), as well as some more advanced functionality such as instance high availability, instance auto scaling, and nested stacks.
  - Heat primarily manages infrastructure, but the templates integrate well with software configuration management tools such as Puppet and Ansible.
  - Operators can customise the capabilities of Heat by installing plugins.

HEAT COMPONENTS

The Orchestration service consists of the following components:

- **heat command-line client**
  - A CLI that communicates with the heat-api to run AWS CloudFormation APIs. End developers can directly use the Orchestration REST API.
- **heat-api component**
  - An OpenStack-native REST API that processes API requests by sending them to the heat-engine over Remote Procedure Call (RPC).
- **heat-api-cfn component**
  - An AWS Query API that is compatible with AWS CloudFormation. It processes API requests by sending them to the heat-engineover RPC.
- **heat-api-cloudwatch component**
  - A CloudWatch-like API service to the heat project.
- **heat-engine**
  - Orchestrates the launching of templates and provides events back to the API consumer.

VNFD

- Vnf descriptor
  - AWS
  - OVF
  - HOT

HOT-DEMO.YML A SINGLE VM HOT

```
heat_template_version: 2015-10-15
description: Launch a basic instance with CirrOS image using the
             ``m1.tiny`` flavor,  and one network.

parameters:
  NetID:
    type: string
    description: Network ID to use for the instance.

resources:
  server:
    type: OS::Nova::Server
    properties:
      image: cirros
      flavor: m1.nano
      networks:
      - network: { get_param: NetID }

outputs:
  instance_name:
    description: Name of the instance.
    value: { get_attr: [ server, name ] }
  instance_ip:
    description: IP address of the instance.
    value: { get_attr: [ server, first_address ] }
```

DESIGN A VNF

```
heat_template_version: 2015-10-15
description: Fake vAPG with CirrOS image using the
             ``m1.nano`` flavor,  and one network.

parameters:
  NetID:
    type: string
    description: Network ID to use for the instance.

resources:
  nodeA:
    type: OS::Nova::Server
    properties:
      image: cirros
      flavor: m1.nano
      networks:
      - network: { get_param: NetID }
  nodeB:
    type: OS::Nova::Server
    properties:
      image: cirros
      flavor: m1.nano
      networks:
      - network: { get_param: NetID }
  diskA:
    type: OS::Cinder::Volume
    properties:
      size: 1
  diskB:
    type: OS::Cinder::Volume
    properties:
      size: 1
  NodeAvolume_attachment:
    type: OS::Cinder::VolumeAttachment
    properties:
      volume_id: { get_resource: diskA }
      instance_uuid: { get_resource: nodeA }
  NodeBvolume_attachment:
    type: OS::Cinder::VolumeAttachment
    properties:
      volume_id: { get_resource: diskB }
      instance_uuid: { get_resource: nodeB }

outputs:
  nodeA:
    description: Name of the instance.
    value: { get_attr: [ nodeA, name ] }
  nodeA_ip:
    description: IP address of the instance.
    value: { get_attr: [ nodeB, first_address ] }
  nodeB:
    description: Name of the instance.
    value: { get_attr: [ nodeA, name ] }
  nodeB_ip:
    description: IP address of the instance
```

LOGS

- Log path:
  - /var/log/<service>
  - /var/log/apache2/*.log (dashboard,keystone)
  - /var/log/syslog

SOME KEYWORDS

- Other features/ techs
  - OpenVSwitch
  - Huge page/ NUMA / CPU-pining
  - Availability-zone / Anti-Affinity
  - vNIC performance: SRIOV, DPDK
  - Commercial SAN(EMC VNX, Scale-IO, Ceph)
  - SDN
  - Customized features (CEE: CMHA)
  - NFV: OPNFV/ MANO etc …
  - Service HA management: Corosync/ Pacemaker
  - API HA/Load balancer: HAPROXY
  - Keepalived
  - Galera
  - Multipath
  - Mirantis Fuel/Ansible/Puppet

REFERENCE/USEFUL URLS

- Installation: OpenStack Installation Guide
- Operation: OpenStack Operations Guide
- API calling with code: Here
- ovs cheat sheet

ERICSSON