OpenStack Networking
（Quantum）
解説

2012年11月9日

OSS基盤技術センター
OSS技術第二課
小田逸郎

- はじめに
- モデルとAPI
- 実装
- その他機能
  - DHCPによるIPアドレス配布
  - L3拡張機能
- 制限事項
- 参考

はじめに

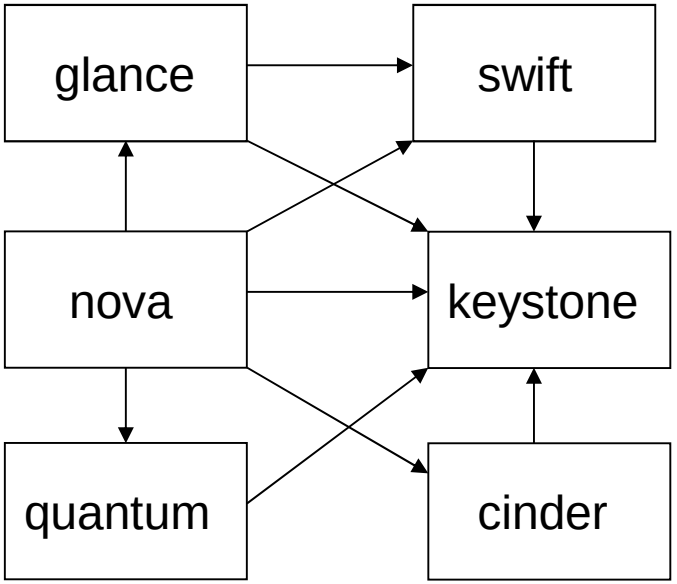
OpenStackのコアプロジェクトのひとつ

OpenStack Compute（Nova）	VM管理
OpenStack Identity（KeyStone）	認証サービス
OpenStack Image Service（Glance）	VMイメージ管理
OpenStack Networking（Quantum）	ネットワーク管理
OpenStack Block Storage Service（Cinder）	ブロックストレージ
OpenStack Object Storage（Swift）	オブジェクトストレージ
OpenStack Dashboard（Horizon）	GUI

プロジェクト間は疎結合

HTTP（REST API）による通信

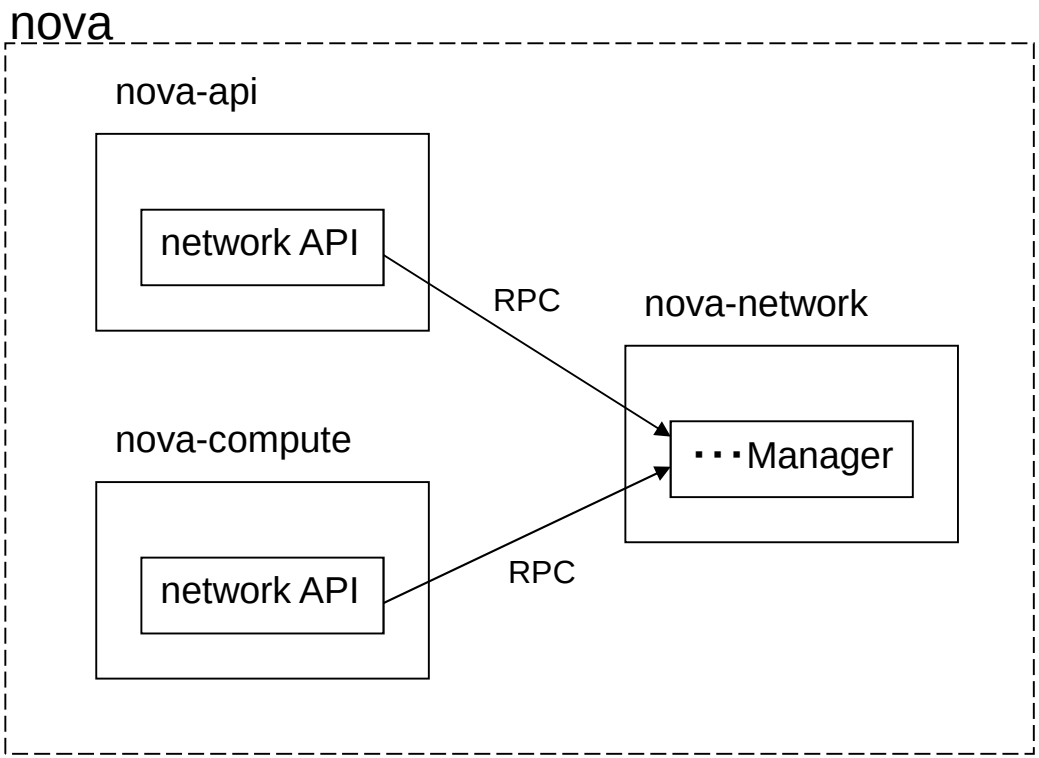
各プロジェクトともOpenStack以外に使用可能



プロジェクト内は、
・通常、複数プロセスで構成
・DBによるプロセス間情報共有、および
・RPCによるプロセス間通信

OpenStackネットワーク管理の構成（cactus以前）

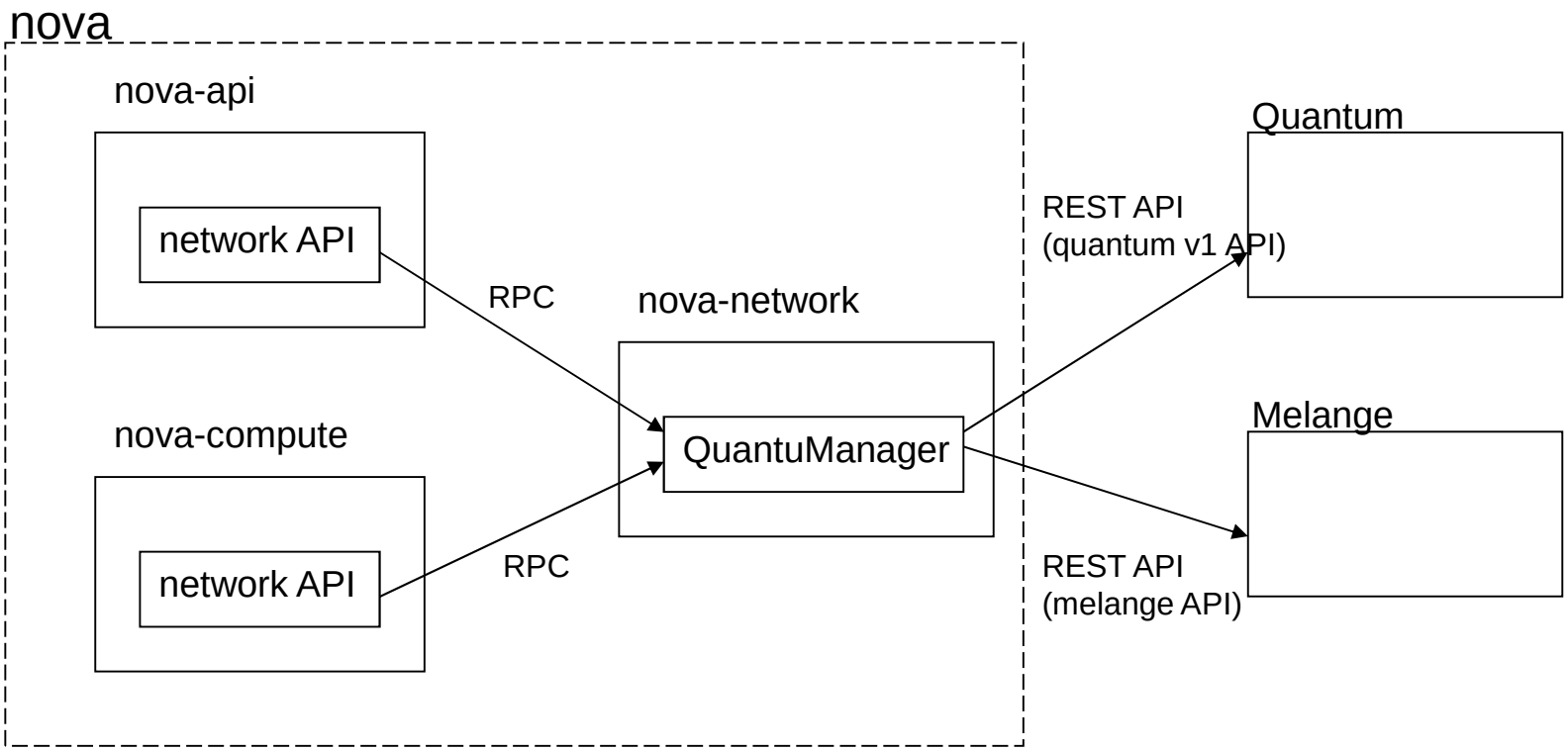
nova-networkプロセスが処理
以下のマネージャを選択可能
・FlatManager： 単一ネットワーク
・FlatDHCPManager： 単一ネットワーク、DHCPでIPアドレス配布
・VlanManger： VLANを使用して、テナント間でネットワークを分離



補足：
nova-networkが動作するホストが、nova-compute間を結ぶネットワークのゲートウェイとなる。

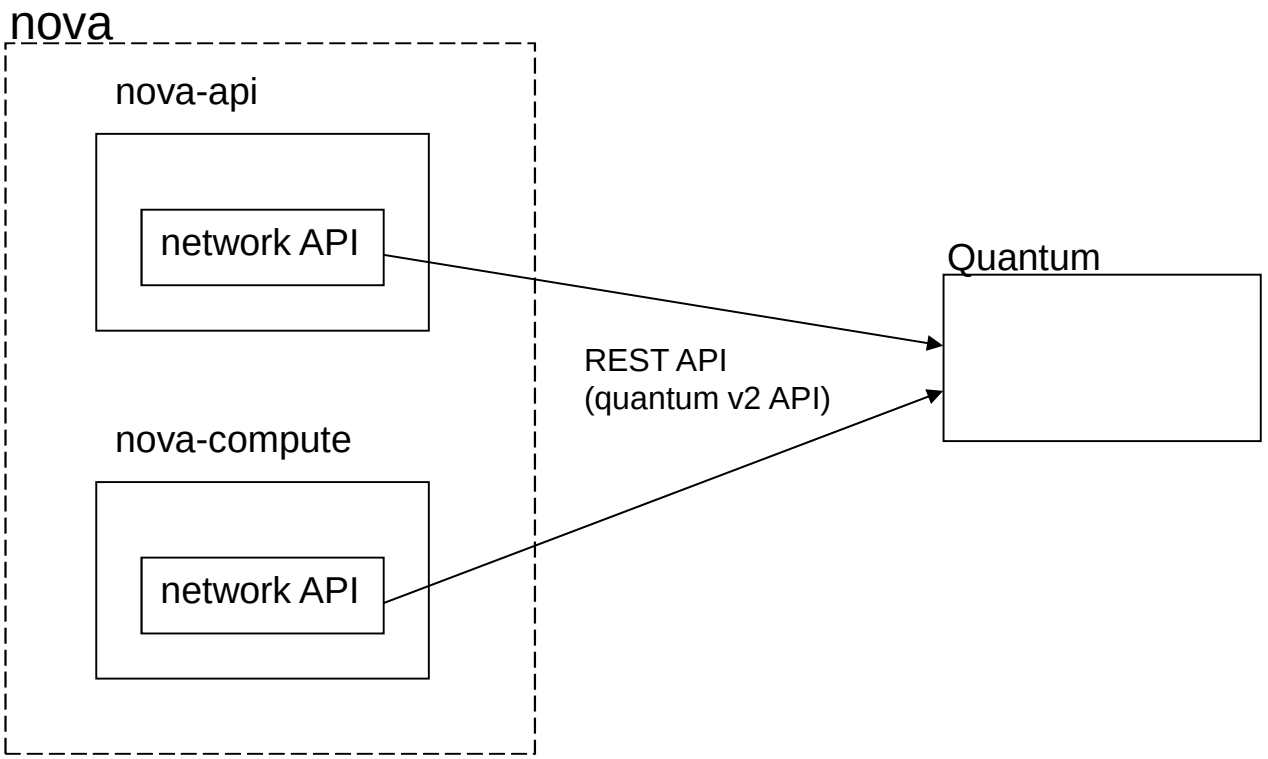
OpenStackネットワーク管理の構成（diablo、essex）

Quantumが登場（実験的な実装。L2管理のみ）
nova-networkにQuantumを使用するマネージャが追加された。
・QuantumManager：　Quantumを使用
IPアドレス管理として、Melangeというプロジェクトができた。



OpenStackネットワーク管理の構成（folsom）

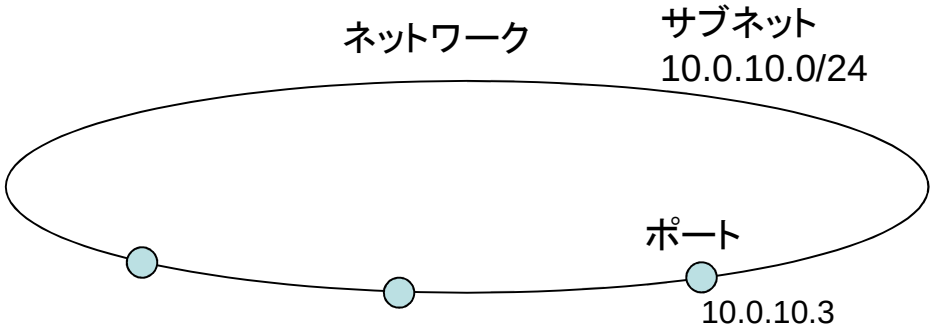
Quantumが正式サポート
IPアドレス管理を取り込み、APIも変わった（quantum v2 API）。
nova-networkは不要となった。（後方互換のため、使用することも可能。
ただし、QuantumManagerはなくなった。）



モデルとAPI

Quantumで扱うリソース

・ネットワーク
　仮想的なL2ネットワーク（L2スイッチ）、L2ブロードキャストの到達範囲。
　テナントの区別あり。
・サブネット
　ネットワーク上のポートに割り当てるIPアドレスブロック（CIDR）
・ポート
　仮想的なL2スイッチ上のポート。ポートとVMのNICを結びつける。



ネットワークAPI

作成（C）	POST	v2.0/networks
更新（U）	PUT	v2.0/networks/ネットワークID
一覧取得（R）	GET	v2.0/networks
情報取得（R）	GET	v2.0/networks/ネットワークID
削除	DELETE	v2.0/networks/ネットワークID

属性：

status（R）	リソースの状態
subnets（R）	サブネットのリスト
name（CUR）	名前
admin_state_up（CUR）	起動状態
tenant_id（CR）	テナントID
shared（CUR）	共有属性
id（R）	ネットワークID

実行例(ネットワーク作成)

入力

```
POST /v2.0/networks HTTP/1.1
Host: 172.17.190.11:9696
Accept: */*
Content-Type: application/json
X-Auth-Token: 24a82ecfa577483ea3af786579a4d41a

{
  "network": {
    "name": "hogehoge",
    "admin_state_up": false,
    "shared": true,
    "tenant_id": "de56b707d3c94686952f8d67a1be0960",
  }
}
```

出力

```
HTTP/1.1 201 Created
Content-Type: application/json; charset=UTF-8
{
  "network": {
    "status": "ACTIVE",
    "subnets": [],
    "name": "hogehoge",
    "admin_state_up": false,
    "tenant_id": "de56b707d3c94686952f8d67a1be0960",
    "shared": true,
    "id": "4cd5e1e1-3a94-4df8-b570-3a904e62c045"
  }
}
```

サブネットAPI

作成（C）	POST	v2.0/subnets
更新（U）	PUT	v2.0/subnets/サブネットID
一覧取得（R）	GET	v2.0/subnets
情報取得（R）	GET	v2.0/subnets/サブネットID
削除	DELETE	v2.0/subnets/サブネットID

属性:

name（CUR）	名前
enable_dhcp（CUR）	DHCP配布の対象とするかどうか
network_id（CR）	ネットワークID
tenant_id（CR）	テナントID
dns_nameservers（CUR）	DNSサーバのリスト
allocation_pools（CUR）	IPアドレス割り当て範囲
host_routes（CUR）	ルーティングテーブルのリスト
ip_version（CR）	IPアドレスバージョン
gateway_ip（CUR）	gatewayアドレス
cidr（CR）	CIDR
id（R）	サブネットID

実行例(サブネット作成)

入力

```
POST /v2.0/subnets HTTP/1.1
Host: localhost:9696
x-auth-token: 051b462fbc744ff9a04b4ebdde1ee8d8
accept: application/json
content-type: application/json
Content-Length: 200

{
  "subnet": {
    "network_id": "e880a8b2-166e-4570-b853-e278704fa453",
    "ip_version": 4,
    "cidr": "10.100.1.0/24"
  }
}
```

出力

```
HTTP/1.1 201 Created
Content-Type: application/json; charset=UTF-8
Content-Length: 377
Date: Mon, 03 Sep 2012 06:47:57 GMT

{
  "subnet": {
    "name": "",
    "enable_dhcp": true,
    "network_id": "e880a8b2-166e-4570-b853-e278704fa453",
    "tenant_id": "9bb11f90f1b54f4da58088b5aabef994",
    "dns_nameservers": [],
    "allocation_pools": [
      {
        "start": "10.100.1.2",
        "end": "10.100.1.254"
      }
    ],
    "host_routes": [],
    "ip_version": 4,
    "gateway_ip": "10.100.1.1",
    "cidr": "10.100.1.0/24",
    "id": "a69ebbd5-f759-4eec-be07-82811b8e8865"
  }
}
```

ポートAPI

作成（C）	POST	v2.0/ports
更新（U）	PUT	v2.0/ports/ポートID
一覧取得（R）	GET	v2.0/ports
情報取得（R）	GET	v2.0/ports/ポートID
削除	DELETE	v2.0/ports/ポートID

属性:

status（R）	リソースの状態
name（CUR）	名前
admin_state_up（CUR）	起動状態
network_id（CR）	ネットワークID
tenant_id（CR）	テナントID
device_owner（CUR）	オーナ
mac_address（CR）	macアドレス
fixed_ips（CUR）	IPアドレスのリスト
id（R）	ポートID
device_id（CUR）	デバイスID

実行例(ポート作成)

入力

```
POST /v2.0/ports HTTP/1.1
Host: 172.17.190.11:9696
Accept: */*
Content-Type: application/json
X-Auth-Token: 6a662b8f9254451abaf6dd19b7502de8

{
  "port": {
    "name": "port1",
    "network_id": "07e5e40e-0be5-4ac7-938d-969f0ee5fda3",
    "admin_state_up": false,
    "mac_address": "ff:ff:ff:ff:ff:05",
    "fixed_ips": [
      {
        "ip_address": "100.0.0.5",
        "subnet_id": "24196f2a-6e5e-4a9a-bbd1-3624f9c096d1"
      }
    ],
    "device_id": "test_device",
    "device_owner": "device owner",
    "tenant_id": "tenant12345"
  }
}
```

出力

```
HTTP/1.1 201 Created
Content-Type: application/json; charset=UTF-8

{
  "port": {
    "status": "ACTIVE",
    "name": "port1",
    "admin_state_up": false,
    "network_id": "07e5e40e-0be5-4ac7-938d-969f0ee5fda3",
    "tenant_id": "tenant12345",
    "device_owner": "device owner",
    "mac_address": "ff:ff:ff:ff:ff:05",
    "fixed_ips": [
      {
        "subnet_id": "24196f2a-6e5e-4a9a-bbd1-3624f9c096d1",
        "ip_address": "100.0.0.5"
      }
    ],
    "id": "27a50f53-f1e9-41ac-a3cb-8e8b1934defb",
    "device_id": "test_device"
  }
}
```

CLI: quantumコマンド

quantum --help	ヘルプ、サブコマンド一覧
quantum help サブコマンド	サブコマンドの詳細文法
quantum net-create	ネットワークの作成
quantum net-update	ネットワークの更新
quantum net-list	ネットワークの一覧
quantum net-show	ネットワークの情報取得
quantum net-delete	ネットワークの削除
quantum subnet-create	サブネットの作成
quantum subnet-update	サブネットの更新
quantum subnet-list	サブネットの一覧
quantum subnet-show	サブネットの情報取得
quantum subnet-delete	サブネットの削除
quantum port-create	ポートの作成
quantum port-update	ポートの更新
quantum port-list	ポートの一覧
quantum port-show	ポートの情報取得
quantum port-delete	ポートの削除

クライアントライブラリ: python-quantumclient
　CLIもこれを利用

CLI実行例

```
$ export OS_USERNAME=admin
$ export OS_PASSWORD=oda
$ export OS_TENANT_NAME=admin
$ export OS_AUTH_URL=http://localhost:5000/v2.0
$ quantum net-create net1
Created a new network:
+-----------------+--------------------------------------+
| Field           | Value                                |
+-----------------+--------------------------------------+
| admin_state_up  | True                                 |
| id              | feabc2cf-b0a5-4a51-a75b-577ccfbe59b4 |
| name            | net1                                 |
| router:external | False                                |
| shared          | False                                |
| status          | ACTIVE                               |
| subnets         |                                      |
| tenant_id       | 7c8deee1fa734054b7bb861ec3922dd9     |
+-----------------+--------------------------------------+
$ quantum subnet-create --name subnet1 net1 10.0.0.0/24
Created a new subnet:
+-----------------+--------------------------------------------------+
| Field           | Value                                            |
+-----------------+--------------------------------------------------+
| allocation_pools | {"start": "10.0.0.2", "end": "10.0.0.254"}      |
| cidr            | 10.0.0.0/24                                      |
| dns_nameservers |                                                  |
| enable_dhcp     | True                                             |
| gateway_ip      | 10.0.0.1                                         |
| id              | ab8ecd17-2693-4371-b468-672b63d18d20             |
| ip_version      | 4                                                |
| name            | subnet1                                          |
| network_id      | feabc2cf-b0a5-4a51-a75b-577ccfbe59b4             |
| tenant_id       | 7c8deee1fa734054b7bb861ec3922dd9                 |
+-----------------+--------------------------------------------------+
```

keystone使用時のおまじない

CLIではIDの代わりに名前を指定可能。
名前はユニークにしておく。（名前はユニーク制約はない。）

```
$ quantum -v port-create --name port1 --fixed-ip subnet_id=subnet1,ip_address=10.0.0.11 net1
...
DEBUG: quantumclient.client REQ: curl -i http://172.17.190.3:9696/v2.0/ports.json -X POST -H "User-
Agent: python-quantumclient" -H "Content-Type: application/json" -H "Accept: application/json" -H  "
X-Auth-Token: 31c9c92cfde848988cb18902e09d4e23"

DEBUG: quantumclient.client REQ BODY: {"port": {"network_id": "feabc2cf-b0a5-4a51-a75b-577ccfbe59b4 "
, "fixed_ips": [{"subnet_id": "ab8ecd17-2693-4371-b468-672b63d18d20", "ip_address": "10.0.0.11"}],  "
name": "port1", "admin_state_up": true}}

DEBUG: quantumclient.client RESP BODY:{"port": {"status": "ACTIVE", "name": "port1", "admin_state_up":
 true, "network_id": "feabc2cf-b0a5-4a51-a75b-577ccfbe59b4", "tenant_id": "7c8deee1fa734054b7bb861ec39
22dd9", "device_owner": "", "mac_address": "fa:16:3e:f8:b0:1a", "fixed_ips": [{"subnet_id": "ab8ecd17-
2693-4371-b468-672b63d18d20", "ip_address": "10.0.0.11"}], "id": "eb50c85a-cb6f-477a-8682-71dfec384ba
3", "device_id": ""}}

Created a new port:
+-----------------+--------------------------------------------------------------------------------+
| Field           | Value                                                                          |
+-----------------+--------------------------------------------------------------------------------+
| admin_state_up  | True                                                                           |
| device_id       |                                                                                |
| device_owner    |                                                                                |
| fixed_ips       | {"subnet_id": "ab8ecd17-2693-4371-b468-672b63d18d20", "ip_address": "10.0.0.11"} |
| id              | eb50c85a-cb6f-477a-8682-71dfec384ba3                                           |
| mac_address     | fa:16:3e:f8:b0:1a                                                              |
| name            | port1                                                                          |
| network_id      | feabc2cf-b0a5-4a51-a75b-577ccfbe59b4                                           |
| status          | ACTIVE                                                                         |
| tenant_id       | 7c8deee1fa734054b7bb861ec3922dd9                                               |
+-----------------+--------------------------------------------------------------------------------+
```

-v オプションでRESTのリクエストとレスポンスを確認

VMの起動

ネットワーク指定

```
$ nova boot --image イメージ名 --nic net-id=ネットワークID サーバ名
```

ポートはnovaが作成。IPアドレス、macアドレスはquantumによる自動割当て。

ポート指定

```
$ nova boot --image イメージ名 --nic port-id=ポートID サーバ名
```

ポートを予め作成しておく。ポート作成時にIPアドレス、macアドレスを指定することもできる。

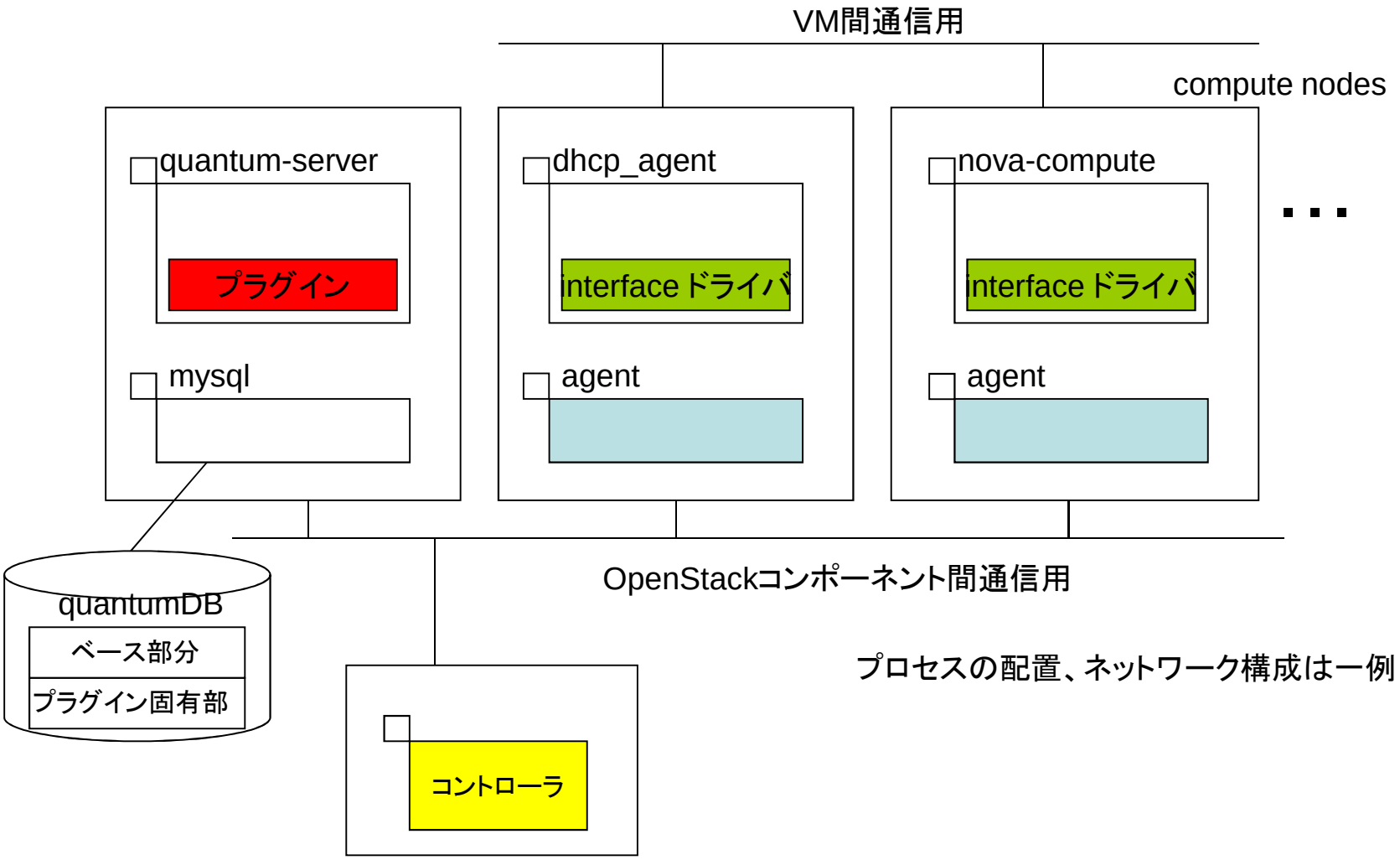
・ --nicオプションを複数指定することにより、複数のNICを装備させることができる。
・いずれの指定方法の場合もVM削除時にnovaにより、ポートは削除される。

実装

プラグイン

- 仮想的なL2環境を実現する処理の実体
- 物理的な環境構成に合わせてプラグインを選択
- マージされているプラグイン
  openvswitch
  linuxbridge
  nicira
  cisco
  nec
  ryu
- 現状は、システムで単一のプラグイン選択しかできない
  （すなわち、ネットワークごとにプラグインを変えることはできない）

プロセス構成



プロセスの配置、ネットワーク構成は一例

各コンポーネントの役割

・プラグイン
- quantum APIの延長で必要な処理を行う。
  ex. - DBにプラグイン固有情報を格納
  - コントローラに必要な情報を伝える
- quantum-serverの一部
  quantum.confで指定

```
core_plugin = quantum.plugins.openvswitch.ovs_quantum_plugin.OVSQuantumPluginV2
```

・interface ドライバ
- 物理的なインタフェースに関する処理（作成、削除など）を行う。
- プラグインに応じて適切なドライバを選択する。
  （プラグイン個別というわけではない。例えば、openvswitchを使用する
  タイプのプラグインは、共通のものが使用できる。）
- nova-computeの一部
  nova.confで指定

```
libvirt_vif_driver = nova.virt.libvirt.vif.LibvirtOpenVswitchDriver
```

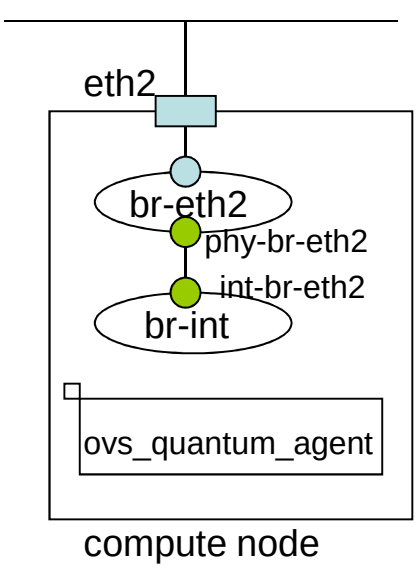
・agent
  - 物理的なインタフェースの作成、削除を検知して、プラグイン固有の
    処理を行う。
  - プラグイン個別のプロセス
    ex. openvswitchプラグインでは、ovs_quantm_agent
  - プラグインによっては、agentがいないタイプのものもある。
    ex. niciraプラグイン

・コントローラ
  - OpenStack（Quantum）の一部ではない。
  - プラグインによっては、外部にコントローラ（ex. OpenFlowコントローラ）
    が存在し、そのコントローラが制御を行う。
    ex. nicira、ryu
    （openvswitchプラグインではコントローラは存在しない。）

処理の流れ

openvswitchプラグインの処理を例に説明

・compute node起動時
ovs_quantum_agent:
必要な初期化を実施。
- br-int(VMの仮想NIC接続用ブリッジ)の初期設定
- 外部接続用ブリッジ(下記図では、br-eth2)の初期設定
ブリッジの作成と物理NIC(下記図ではeth2)の接続は予めしておく必要あり。
- 上記ブリッジ間の接続



ovs_quantum_plugin.ini設定例

```
tenant_network_type = vlan
network_vlan_ranges = default:100:199
bridge_mappings = default:br-eth2
```

補足:
本例は、network typeがvlanの場合を説明。
network typeは、gre というのもある。ホスト間の通信で
GREトンネリングを使用。ホスト間ネットワークの設定や、
agentの処理は本例と異なる。

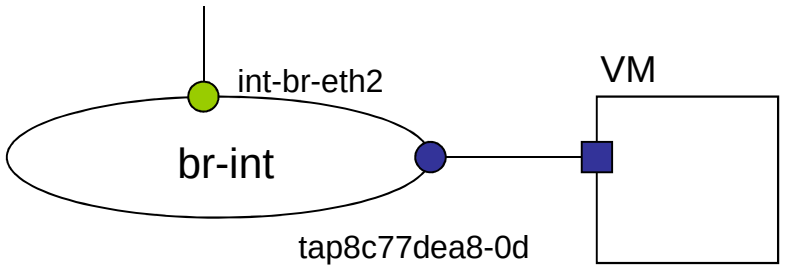
・ネットワーク作成時（quantum API create network時）
プラグイン:
ネットワークIDとvlan-idの対応をつけ、DBに記録

```
$ quantum net-create net1
Created a new network:
+---------------------------+--------------------------------------+
| Field                     | Value                                |
+---------------------------+--------------------------------------+
| admin_state_up            | True                                 |
| id                        | fa2ba33b-b50f-47c2-8c41-ddca7a481dc6 |
| name                      | net1                                 |
| provider:network_type     | vlan                                 |
| provider:physical_network | default                              |
| provider:segmentation_id  | 100                                  |
| router:external           | False                                |
| shared                    | False                                |
| status                    | ACTIVE                               |
| subnets                   |                                      |
| tenant_id                 | 4d447d1234fb47feaebf657ea876a845     |
+---------------------------+--------------------------------------+
```

プラグイン固有属性

ネットワーク net1 は、vlan id 100で、ノード間の通信を行う。

・VM作成時
  novaのinterfaceドライバ:
  VMの仮想NICに対応するtapデバイスを作成し、br-intに接続する。
  後でagentで参照するため、Interfaceテーブルに必要な情報を設定しておく。
  - external-ids:iface-id　にポートID
  - external-ids:attached-mac にmacアドレス



```
$ ifconfig
...
tap8c77dea8-0d Link encap:Ethernet  HWaddr 1a:fe:53:16:54:45
          inet6 addr: fe80::18fe:53ff:fe16:5445/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:373 errors:0 dropped:0 overruns:0 frame:0
          TX packets:179 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:500
          RX bytes:68692 (68.6 KB)  TX bytes:35044 (35.0 KB)

$ sudo ovs-vsctl list-ports br-int
int-br-eht2
tap8c77dea8-0d

$ sudo ovs-vsctl list Interface
_uuid               : d13469a0-a5f6-4b36-bed6-709c22e82e1b
admin_state         : up
...
external_ids        : {attached-mac="fa:16:3e:b5:54:65",
iface-id="8c77dea8-0d25-4dde-9165-9d2595cd9375", iface-status=active,
 vm-uuid="a66595e0-b318-49ae-98d5-564b7b42bfbf"}
ingress_policing_burst: 0
ingress_policing_rate: 0
lacp_current        : []
link_resets         : 1
link_speed          : 10000000
link_state          : up
mac                 : []
mtu                 : 1500
name                : "tap8c77dea8-0d"
ofport              : 11
...
```

interfaceの作成、
ovsへの接続、
ovsのInterfaceテーブルへの
情報設定は、
novaのinterfaceドライバが実施

・物理インタフェース検出時

ovs_quantum_agent:

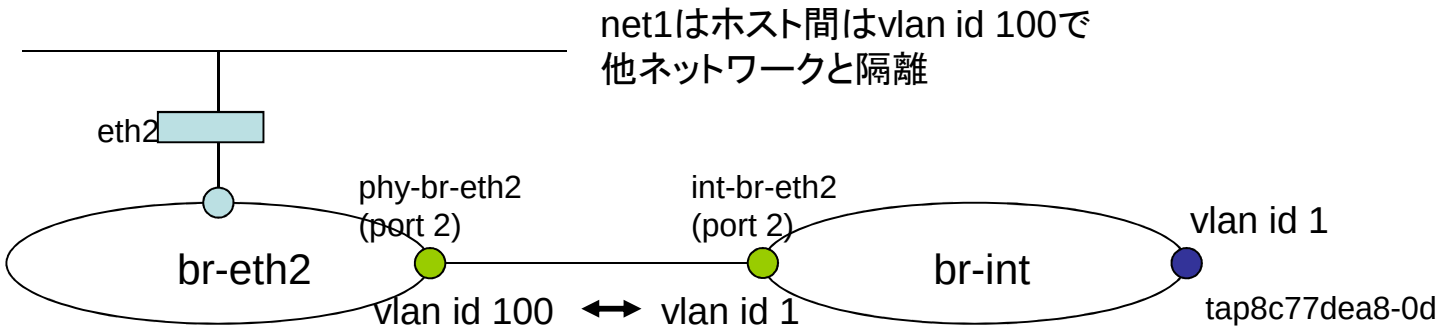
- br-intを定期的に監視し、ポートができたことを検出
- quantum-serverよりポートおよびネットワーク情報を取得（RPC使用）
  ネットワークに割り当てられたvlan idが分かる。
- 目的のovs portにvlan tagを設定。
  ホスト内のネットワーク隔離には、ホスト内でvlan idを割り当て。
  ホスト内のovs_quantum_agentがネットワークのvlan idとの対応を管理。
- br-int、br-eth2間のフローテーブルを設定
  ホスト内vlan idとネットワークvlan idの変換を行う。

```
$ sudo ovs-vsctl list Port
_uuid               : c0d9d297-2bee-411f-9bfc-b6c3ff767156
bond_downdelay      : 0
bond_fake_iface     : false
bond_mode           : []
bond_updelay        : 0
external_ids        : {}
fake_bridge         : false
interfaces          : [d13469a0-a5f6-4b36-bed6-709c22e82e1b]
lacp                : []
mac                 : []
name                : "tap8c77dea8-0d"
other_config        : {}
qos                 : []
statistics          : {}
status              : {}
tag                 : 1
trunks              : []
vlan_mode           : []
...
```

ovs_quantum_agentは、
ovsにportが追加されたことを
検出し、tagを設定。
（tag == vlan id
同じtagのポートとしか通信できない。
tagを設定するまではどことも通信
できない。）

```
$ sudo ovs-ofctl dump-flows br-int
NXST_FLOW reply (xid=0x4):
 cookie=0x0, duration=2009.375s, table=0, n_packets=6, n_bytes=468, priority=2,in_port=2 actions=drop
 cookie=0x0, duration=177.705s, table=0, n_packets=0, n_bytes=0, priority=3,in_port=2,dl_vlan=100
 actions=mod_vlan_vid:1,NORMAL
 cookie=0x0, duration=2009.659s, table=0, n_packets=9, n_bytes=1434, priority=1 actions=NORMAL

$ sudo ovs-ofctl dump-flows br-eth2
NXST_FLOW reply (xid=0x4):
 cookie=0x0, duration=1997.043s, table=0, n_packets=6, n_bytes=468, priority=2,in_port=2 actions=drop
 cookie=0x0, duration=165.4s, table=0, n_packets=9, n_bytes=1454, priority=4,in_port=2,dl_vlan=1
 actions=mod_vlan_vid:100,NORMAL
 cookie=0x0, duration=1997.312s, table=0, n_packets=0, n_bytes=0, priority=1 actions=NORMAL
```

ovs_quantum_agentは、br-int、br-eth2間でvlan idを変換するようにフローテーブルを設定。



net1は当該ホスト内はvlan id 1で
他ネットワークと隔離
（別ホストではidは違うかもしれない）

プラグインの実装（OpenFlowコントローラ型）

compute node上のopenvswitchをOpenFlowスイッチとして、制御するタイプ

・プラグイン
リソースの作成時、コントローラに情報を設定したり、DBに必要な情報を格納。ネットワークだけでなく、ポート作成時も。

・interfaceドライバ
共通のopenvswich用ドライバを使用することが多い。
独自のドライバを実装し、コントローラとの通信を行うことも考えられる。

・agent
物理interfaceの検出時、コントローラに通知し、コントローラがフローテーブルを設定する。
どのマシンのopenvswitchにフローテーブルを設定するかを教えるのがagentの役目。
openvswitchにコントローラの設定とフローテーブルの初期設定をしておき、VMの最初のパケットが流れたときに検出することにより、agentを置かないことも考えられる。

その他の機能

DHCPによるIPアドレスの配布

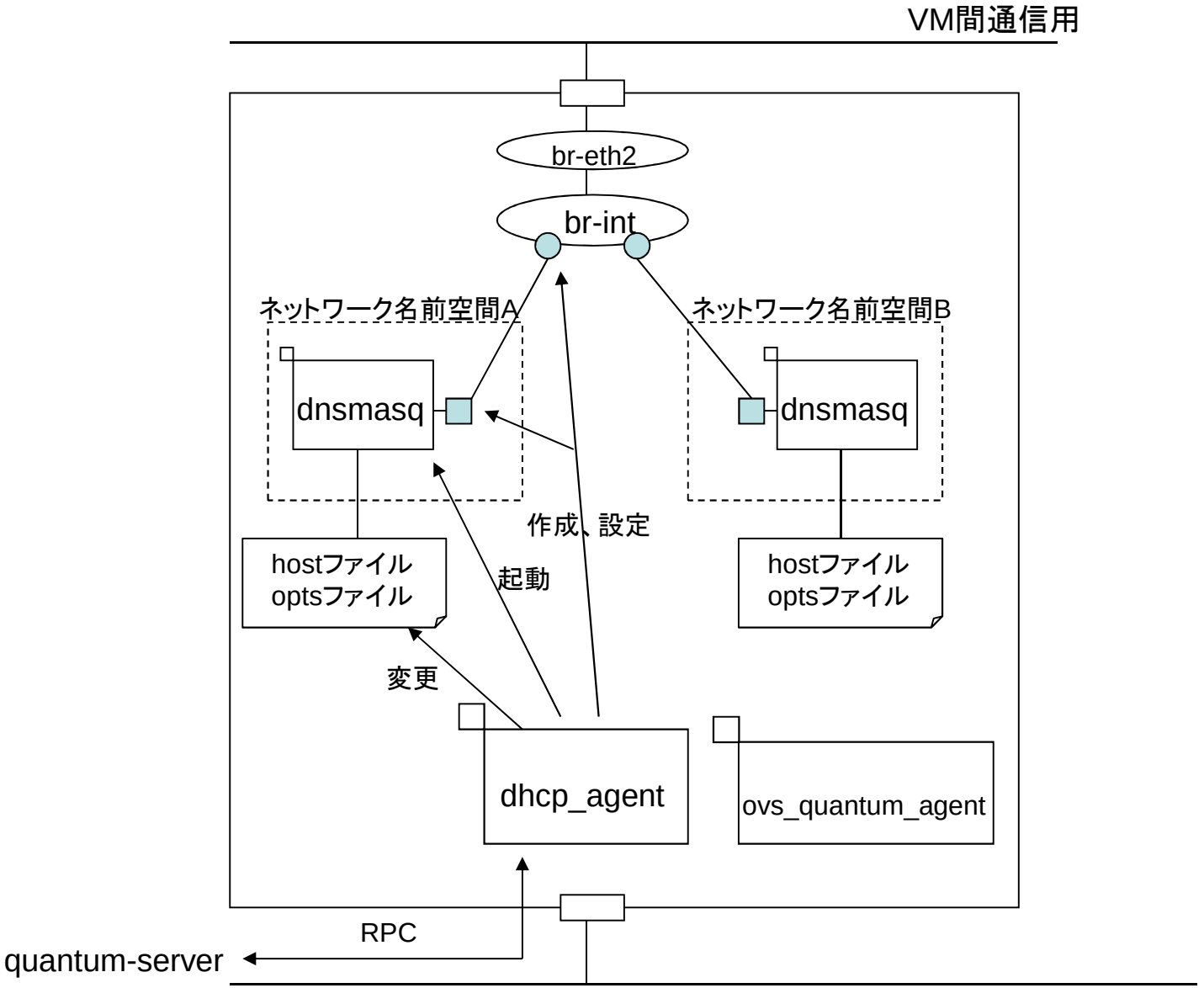
dhcp_agentにより実現

処理の流れ

・サブネットの作成を契機とし、dnsmasq用ポートの作成と、openvswitchへの接続を行う。novaと同様、プラグインに対応したinterfaceドライバを指定しておく。（dhcp_agent.iniに設定）
・サブネットごとにdnsmasqを起動する。個々のdnsmasqは独立したネットワーク名前空間で実行する。（対応するinterfaceも同じ名前空間に作成する）
・ポートの作成を契機とし、dnsmasqのhostファイル、optsファイルを書き換え、dnsmasqに通知する。
・quantum-serverからRPCを通して、サブネットの作成、削除、ポートの作成、削除の通知がdhcp_agentに行く。dhcp_agentからquantum-serverへの情報取得もRPCを使用。（DBを参照する設定も可）

課題：スケーラビリティ

・すべてのサブネットをひとつのdhcp_agentで処理
dhcp_agentはひとつのマシンでしか動作できない。
（アクティブ・スタンバイのHA構成を取ることは可能）
・サブネットごとにdnsmasqを起動

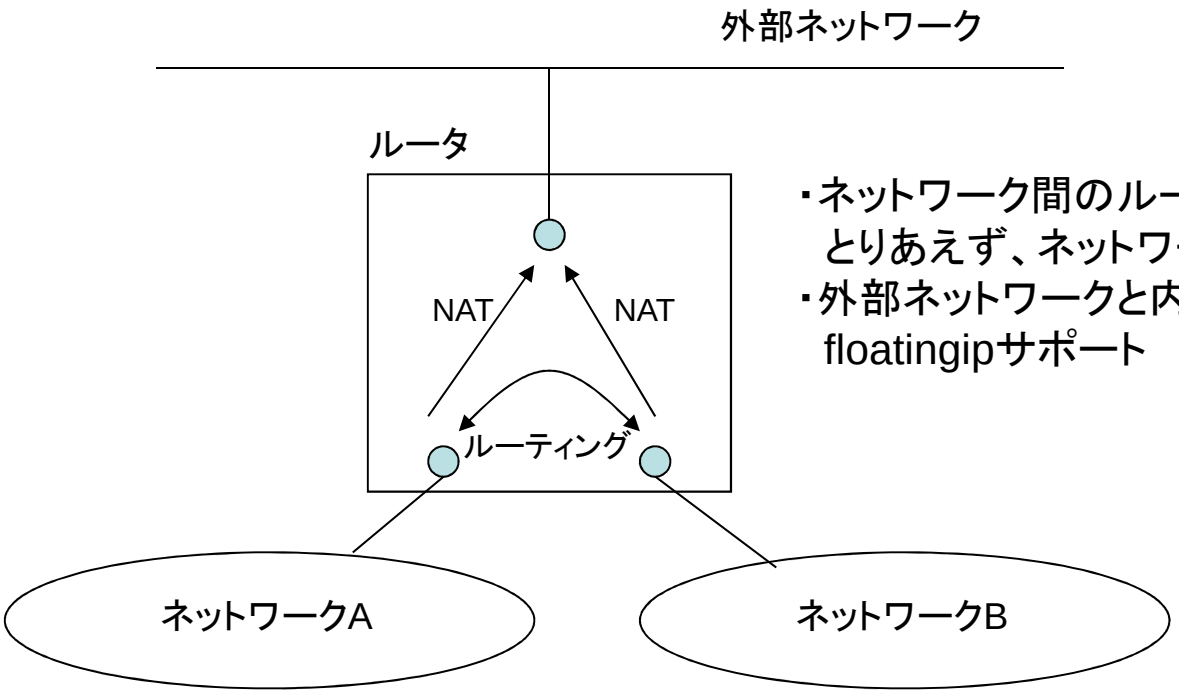

VM間通信用
br-eth2
br-int
ネットワーク名前空間A
ネットワーク名前空間B
dnsmasq
dnsmasq
作成、設定
起動
hostファイル
optsファイル
hostファイル
optsファイル
変更
dhcp_agent
ovs_quantum_agent
RPC
quantum-server

```
$ ip netns list
qdhcp-c6943641-2937-4e68-b010-53e14002d954

$ sudo ip netns exec qdhcp-c6943641-2937-4e68-b010-53e14002d954 ifconfig
lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

ns-5f9c44ba-a5 Link encap:Ethernet  HWaddr fa:16:3e:9a:8a:14
          inet addr:10.0.0.2  Bcast:10.0.0.255  Mask:255.255.255.0
          inet6 addr: fe80::f816:3eff:fe9a:8a14/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:601 errors:0 dropped:0 overruns:0 frame:0
          TX packets:327 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:124808 (124.8 KB)  TX bytes:64792 (64.7 KB)
```

- ネットワークが異なれば、サブネットのCIDRは重なってもよいので、dnsmasqはサブネットごとに独立したネットワーク名前空間で動作させる必要がある。
- ネットワーク名前空間
  IPアドレス、インタフェース、iptables、ルーティングテーブルが独立
- Linuxの機能

L3拡張機能

・追加されたリソース
  ルータ
  フローティングIP
・実験的な実装であり、正式なドキュメントなし。
  将来的に変更される可能性あり



・ネットワーク間のルーティング
  とりあえず、ネットワーク間がつながるだけ
・外部ネットワークと内部ネットワーク間のNAT、
  floatingipサポート

ルータAPI

作成（C）	POST	v2.0/routers
更新（U）	PUT	v2.0/routers/ルータID
一覧取得（R）	GET	v2.0/routers
情報取得（R）	GET	v2.0/routers/ルータID
削除	DELETE	v2.0/routers/ルータID
内部ネットワーク追加	PUT	v2.0/routers/ルータID/add_router_interface
内部ネットワーク削除	PUT	v2.0/routers/ルータID/remove_router_interface

属性:

status（R）	リソースの状態
external_gateway_info（CUR）	外部ネットワーク情報
name（CUR）	名前
admin_state_up（CUR）	起動状態
tenant_id（CR）	テナントID
id（R）	ルータID

実行例(ルータ作成)

入力

```
POST /v2.0/routers HTTP/1.1
X-Auth-Token: 89e9a95d45f2436e80969ce3c57fa9ef
Accept: application/json
Content-Type: application/json
Content-Length: 32

{
  "router": {
    "name": "router-1"
  }
}
```

出力

```
HTTP/1.1 201 Created
Content-Type: application/json
Content-Length: 202

{
  "router": {
    "status": "ACTIVE",
    "external_gateway_info": null,
    "name": "router-1",
    "admin_state_up": true,
    "tenant_id": "5af5eae40e4c4d73875a535994886627",
    "id": "622e97ee-3483-446f-9e6e-462e01e5e846"
  }
}
```

フローティングIP API

作成（C）	POST	v2.0/floatingips
更新（U）	PUT	v2.0/floatingips/フローティングIPID
一覧取得（R）	GET	v2.0/floatingips
情報取得（R）	GET	v2.0/floatingips/フローティングIPID
削除	DELETE	v2.0/floatingips/フローティングIPID

属性:

router_id（R）	ルータID
tenant_id（CR）	テナントID
floating_network_id（CR）	外部ネットワークID
fixed_ip_address（CUR）	固定IPアドレス
floating_ip_address（R）	フローティングIPアドレス
port_id（CUR）	ポートID
id（R）	フローティングIP ID

実行例(フローティングIP作成)

入力

```
POST /v2.0/floatingips HTTP/1.1
Host: 172.17.190.21:9696
Content-Length: 79
x-auth-token: 41b949d08c474de38e7387ccd4634514
content-type: application/json
accept: application/json

{
  "floatingip": {
    "floating_network_id": "7c027a22-9fc9-4f42-9eed-e32d05a24cba"
  }
}
```

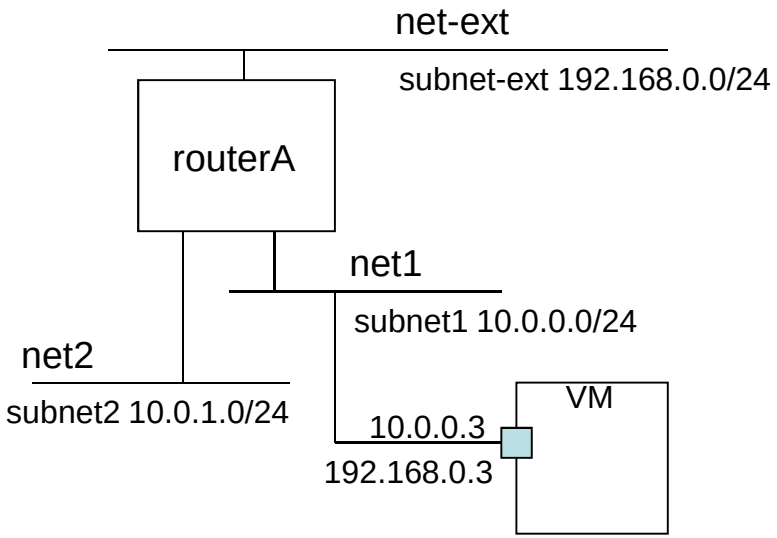
出力

```
HTTP/1.1 201 Created
Content-Type: application/json
Content-Length: 273

{
  "floatingip": {
    "router_id": null,
    "tenant_id": "58d93fe6a1ad40f1897a27abfbeab35e",
    "floating_network_id": "7c027a22-9fc9-4f42-9eed-e32d05a24cba",
    "fixed_ip_address": null,
    "floating_ip_address": "10.200.1.2",
    "port_id": null,
    "id": "8f86f823-cf0c-48bb-8f14-2c9c9713f031"
  }
}
```

CLI: quantumコマンド

router-create	ルータの作成
router-delete	ルータの削除
router-gateway-clear	外部ネットワーク切断
router-gateway-set	外部ネットワーク接続
router-interface-add	内部ネットワーク追加
router-interface-delete	内部ネットワーク削除
router-list	ルータの一覧
router-show	ルータの情報取得
router-update	ルータの更新
floatingip-associate	フローティングIPと固定IPの関係付け
floatingip-create	フローティングIPの作成
floatingip-delete	フローティングIPの削除
floatingip-disassociate	フローティングIPと固定IPの関係解除
floatingip-list	フローティングIPの一覧
floatingip-show	フローティングIPの情報取得

実行例



ネットワークとルータの作成、コマンド列のみ

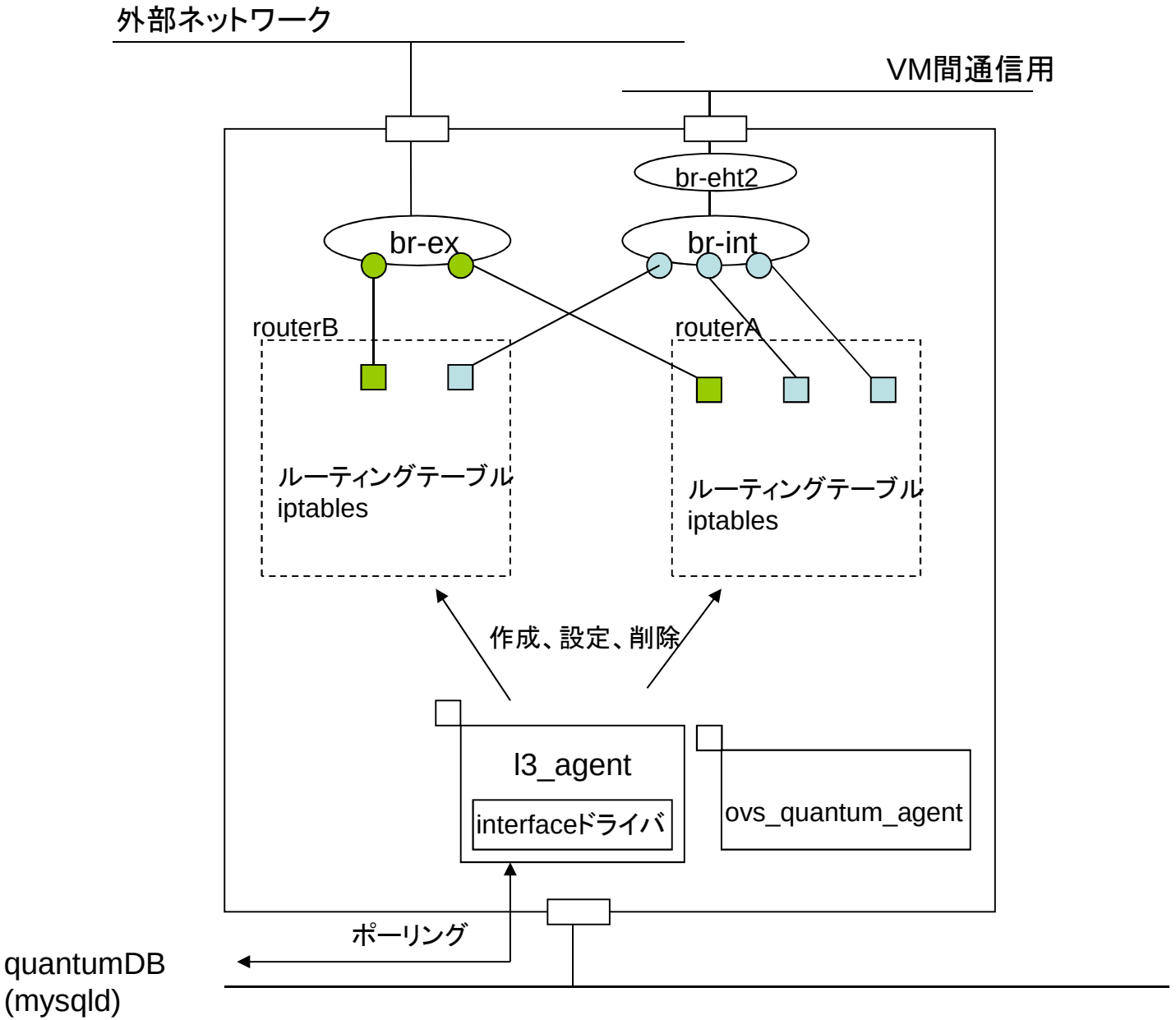
```
$ quantum net-create net1
$ quantum subnet-create --name subnet1 net1 10.0.0.0/24
$ quantum net-create net2
$ quantum subnet-create --name subnet2 net2 10.0.1.0/24
$ quantum net-create net-ext -- --router:external=True
$ quantum subnet-create --name subnet-ext --gateway 192.168.0.1 net-ext 192.168.0.0/24 -- --enable_dhcp=False
$ quantum router-create routerA
$ quantum router-gateway-set routerA net-ext
$ quantum router-interface-add routerA subnet1
$ quantum router-interface-add routerA subnet2
```

フローティングIPの割り当て

```
$ VMの作成
$ quantum floatingip-create net-ext
Created a new floatingip:
+---------------------+--------------------------------------+
| Field               | Value                                |
+---------------------+--------------------------------------+
| fixed_ip_address    |                                      |
| floating_ip_address | 192.168.0.3                          |
| floating_network_id | fd302423-d3f8-4f3b-bc14-f67c7e97e64e |
| id                  | 17f97c24-13c7-4b82-91c4-a143e2d463c3 |
| port_id             |                                      |
| router_id           |                                      |
| tenant_id           | 7c8deee1fa734054b7bb861ec3922dd9     |
+---------------------+--------------------------------------+
$ quantum port-list
VMのNIC(10.0.0.3)のポートIDを探す
$ quantum floatingip-associate 17f97c24-13c7-4b82-91c4-a143e2d463c3 8c77dea8-0d25-4dde-9165-9d2595cd9375
Associated floatingip 17f97c24-13c7-4b82-91c4-a143e2d463c3
$ quantum floatingip-show 17f97c24-13c7-4b82-91c4-a143e2d463c3
+---------------------+--------------------------------------+
| Field               | Value                                |
+---------------------+--------------------------------------+
| fixed_ip_address    | 10.0.0.3                             |
| floating_ip_address | 192.168.0.3                          |
| floating_network_id | fd302423-d3f8-4f3b-bc14-f67c7e97e64e |
| id                  | 17f97c24-13c7-4b82-91c4-a143e2d463c3 |
| port_id             | 8c77dea8-0d25-4dde-9165-9d2595cd9375 |
| router_id           | 16a7d4e1-d3e0-4aec-bbbd-d32b6d14448d |
| tenant_id           | 7c8deee1fa734054b7bb861ec3922dd9     |
+---------------------+--------------------------------------+
```

L3拡張機能の実装

・l3_agentプロセスが処理を行う。
・DBを監視（ポーリング）し、状態が変更されたら処理を行う。
・ルータ作成時
ルータ用のネットワーク名前空間を作成。ルータはルータごとの個別の
ネットワーク名前空間を持つ。
・外部ネットワーク接続時
外部ネットワーク用のポート作成、インタフェース作成、openvswitch（br-ex）
への接続などを行う。nova、dhcp_agentと同様にプラグインに応じた
interfaceドライバを設定しておく。（l3_agent.ini）
・内部ネットワーク接続時
内部ネットワーク用のポート作成、インタフェース作成、openvswitch（br-int）
への接続などを行う。ルーティングテーブル、iptablesの設定を行う。
・フローティングIPが固定IPに関係付けられたとき
iptables（NAT）の設定を行う。
・ルータの実体は、Linux上の独立したネットワーク名前空間で定義された
インタフェース、ルーティングテーブル、iptables。


外部ネットワーク
VM間通信用
br-eht2
br-ex
br-int
routerB
routerA
ルーティングテーブル
iptables
ルーティングテーブル
iptables
作成、設定、削除
l3_agent
interfaceドライバ
ovs_quantum_agent
ポーリング
quantumDB
(mysqld)

```
$ ip netns list
qdhcp-c6943641-2937-4e68-b010-53e14002d954
qrouter-5e37b0c6-9f85-40cd-9d06-fb6f814448e6

$ sudo ip netns exec qrouter-5e37b0c6-9f85-40cd-9d06-fb6f814448e6 ifconfig
lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
...
qg-f1e3b587-81 Link encap:Ethernet  HWaddr fa:16:3e:47:f1:9b
          inet addr:192.168.0.2  Bcast:192.168.0.255  Mask:255.255.255.0
...
qr-e84c655e-8a Link encap:Ethernet  HWaddr fa:16:3e:1f:7e:7a
          inet addr:10.0.0.1  Bcast:10.0.0.255  Mask:255.255.255.0
...
qr-f03226bf-a4 Link encap:Ethernet  HWaddr fa:16:3e:61:4b:81
          inet addr:10.0.1.1  Bcast:10.0.1.255  Mask:255.255.255.0
...

$ sudo ip netns exec qrouter-5e37b0c6-9f85-40cd-9d06-fb6f814448e6 route -n
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0         192.168.0.1     0.0.0.0         UG    0      0        0 qg-f1e3b587-81
10.0.0.0        0.0.0.0         255.255.255.0   U     0      0        0 qr-e84c655e-8a
10.0.1.0        0.0.0.0         255.255.255.0   U     0      0        0 qr-f03226bf-a4
192.168.0.0     0.0.0.0         255.255.255.0   U     0      0        0 qg-f1e3b587-81

$ sudo ip netns exec qrouter-5e37b0c6-9f85-40cd-9d06-fb6f814448e6 iptables -L -t nat
...
DNAT       all  --  anywhere             192.168.0.3          to:10.0.0.3
...
SNAT       all  --  10.0.0.0/24          anywhere             to:192.168.0.2
SNAT       all  --  10.0.1.0/24          anywhere             to:192.168.0.2
...
```

名前空間

interface

net-ext

net1

net2

ルーティング
テーブル

NAT

制限事項

制限事項

Novaでできていて、Quantumでできていないこと

セキュリティグループ
・実体は、nova-computeノード上のiptables
nova-computeの機能であり、今でも使えることは使える。
・単一IPアドレスブロックを仮定
・openvswitchを使用すると、iptablesは利かない。
この点に関しては、別のinterfaceドライバ（LibvirtHybridOVSBridgeDriver）を使用することにより回避できる。ただし、単一IPアドレスブロックの仮定については解消できない。
・ファイアウォール機能については、grizzlyに向けて、議論が進められている。

参考

情報源

- Launchpad
  https://launchpad.net/quantum
  プロジェクトの状況を掴む入り口。ブループリントやバグレポートなど。

- ソースコード
  https://github.com/openstack/quantum

- ドキュメント
  Quantum API
  http://docs.openstack.org/api/openstack-network/2.0/content/