OpenStack Neutron LbaaS integration with physical F5 in OpenContrail SDN

Chase Abbott, 2015-03-12

Republished from TCP Cloud on request from the project authors, Radovan Gibala and Filip Kolar at F5.

In this blog we would like to show how to integrate physical F5 under OpenContrail SDN and create Load Balancer pools through standard Neutron LbaaS API.

Load Balancers are very important part of cloud and OpenStack Neutron has enabled to use LbaaS features since release Grizzly. However upstream implementation with OpenvSwitch/HAProxy does not provide high availability by design. SDN OpenContrail provides HA LbaaS feature with HAProxy from release IceHouse and for example Symantec comes with great performance results.(http://www.slideshare.net/RudrajitTapadar/meetup-vancouverpptx-1)

However lots of companies still need to use physical load balancers especially F5 Networks for performance (HW SSL offloading) and other feature benefits. Therefore integration with physical load balancers is mandatory. Second mandatory requirement is tight integration with Neutron LbaaS to enable developers manage different LbaaS providers through standard API and orchestrate infrastructure by OpenStack Heat.

There exist different SDN solution, which support integration with physical F5, but none can provide it thru Neutron LbaaS API. They usually offer possibility to manage F5 in their own administrator dashboard, which does not provide the real benefits of automation. OpenContrail as only one SDN/NFV solution released a new driver for physical and virtual F5 balancers, which is compliant with previous two requirements.

In this blog we show:

- How to configure OpenContrail to use F5 driver
- How to provisioning physical F5 thru Neutron LbaaS API
- How to automatically orchestrate them via OpenStack Heat

Lab Overview

OpenContrail 2.20 contains beta release for managing physical or virtual F5 through OpenStack Neutron LbaaS API. OpenStack Neutron LbaaS v1 contains following objects and their dependencies: member, pool, VIP, monitor.
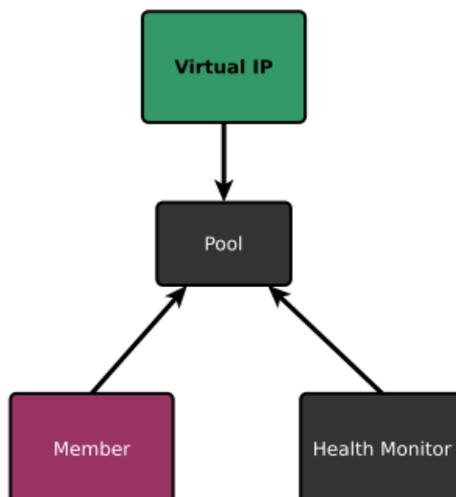

F5 can operate now only in "global routed mode", where all the VIPs are assumed to be routable from clients and all members are routable from F5 device. Therefore the entire configuration on F5 for L2 and L3 must be pre-provisioned.

In the global routed mode, because all access to and from the F5 device is assumed to globally routed, there is no segregation between tenant services on F5 device possible. In other words, overlapping addresses across tenants/networks is not a valid configuration.

Following assumptions made for global routed mode of F5 LBaaS support:

- All tenant networks are in the same namespace as fabric corporate network
- IP Fabric is also in the same namespace as corporate network
- All VIPs are also in the same namespace as tenant/corporate networks
- F5 could be attached to corporate network or to IP Fabric
- The following network diagram capture lab topology, where we tested F5 integration
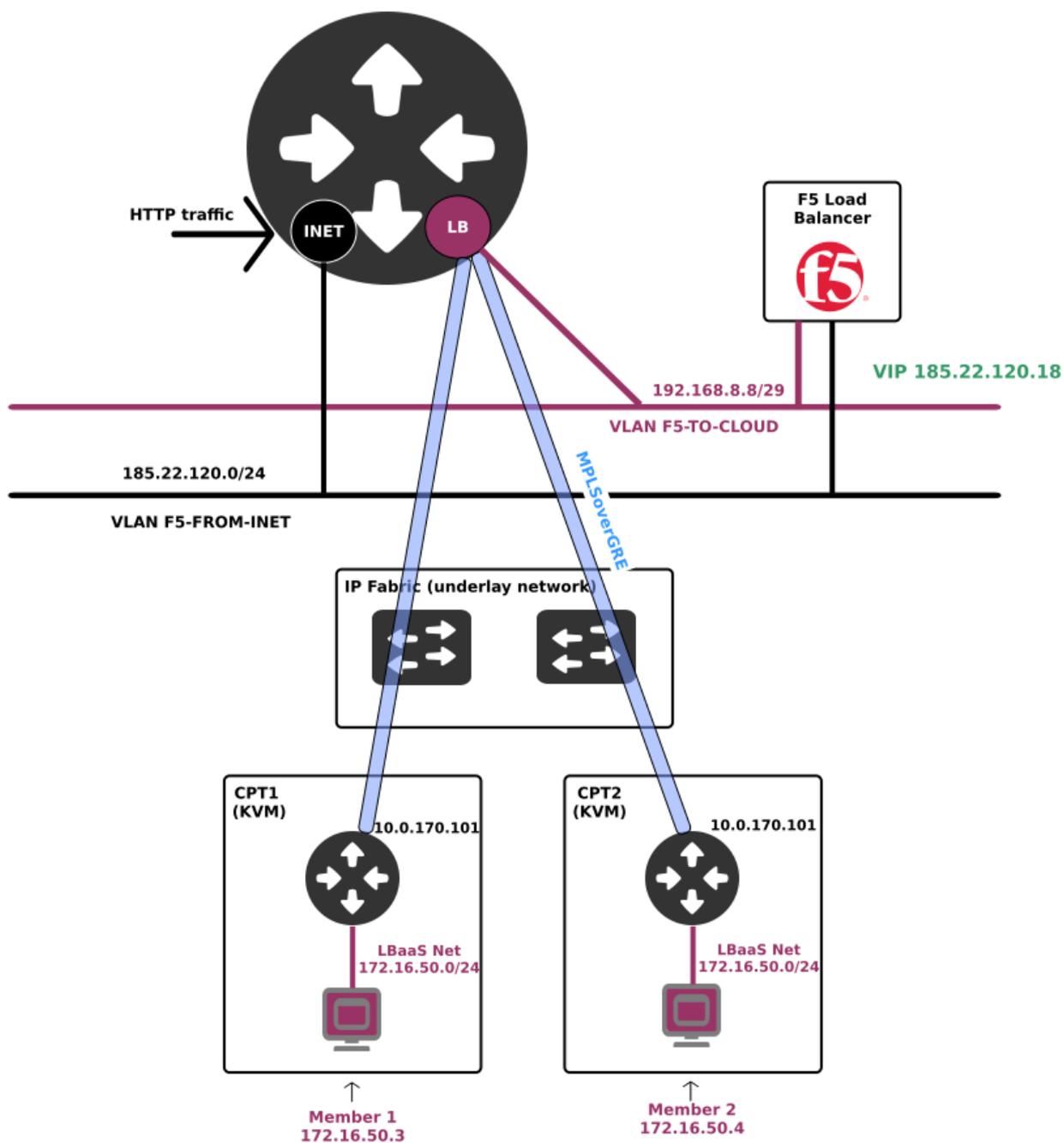
- **VLAN F5-FROM-INET 185.22.120.0/24** - VLAN with public IP addresses used for VIP on F5 load balancer
- **VLAN F5-TO-CLOUD 192.168.8.8/29** - VLAN between F5 and Juniper MX LB VRF (subinterface). It is transport network used for communication between members and F5
- **Underlay network 10.0.170.0/24** - underlay internal network for OpenContrail/OpenStack services (iBGP peering, MPLSoverGRE termination on Juniper MX). Each compute node (vRouter) and Juniper MX have IP addresses from this subnet
- **VIP network 185.22.120.0/24** - used for VIP pool. Same network as F5-FROM-INET, but created as VN in

Neutron. Neutron LbaaS VIP cannot be created from network, which does not exist in OpenStack

- **Overlay Member VN (Virtual Network) 172.16.50.0/24** - Standard OpenStack Neutron network with Route Target into LB routing-instance (VRF) on Juniper MX. This network is propagated into LB VRF

Initial configuration on F5

- preconfigured VLANs on specific ports with appropriate Self IPs. F5 must be able to access members in OpenStack cloud and INET for VIP pool.
- accessible management from OpenContrail controllers

Initial configuration on Juniper MX (DC Gateway)

- In this case configuration for MX is manual, so there must be preconfigured VRF for LB and INET
- Static routes must be configured correctly

INITIAL OPENCONTRAIL CONFIGURATION

OpenContrail 2.20 contains two new components, which are responsible for managing F5:

- contrail-f5 - package with Big IP interface for f5 load balancer.
- f5_driver.py - driver itself delived in package contrail-config-openstack.

We need to create service appliance set definition for general F5 balancer and service appliance for one specific F5 device. These configuration enables to use F5 as LbaaS provider in Neutron API.

Service Appliance Set as LBaaS Provider

In neutron, loadbalancer provider is statically configured in neutron.conf using following parameter:

```
[service_providers]service_provider= LOADBALANCER:Opencontrail:neutron_plugin_contrail.plugins.openco
```

In OpenContrail, neutron LBaaS provider is configured using configuration object "service appliance set". This config object includes "python" module to load for LBaaS driver. All the configuration knobs of the LBaaS driver is populated to this object and passed to the driver.

OpenContrail F5 driver options in current beta version:

- device_ip - ip address for management configuration of F5
- sync_mode – replication
- global_routed_mode - only one mode, which is now supported
- ha_mode - standalone is default settings
- use_snat - use F5 for SNAT
- vip_vlan - vlan name on F5, where vip subnet is routed. Our case is F5-TO-INET
- num_snat – 1
- user - admin user fo connection to F5
- password - password for admin user to F5
- MX parameters - (mx_name, mx_ip, mx_f5_interface, f5_mx_interface) are used for dynamic provisioning routing instances (VRF) between Juniper MX and F5. We have not tested this feature with F5 driver yet.

At first there must be installed contrail-f5 and python-suds packages. After that create service_appliance_set for neutron lbaas provider F5.

```
apt-get install python-suds contrail-f5
/opt/contrail/utils/service_appliance_set.py --api_server_ip 10.0.170.30 --api_server_port 8082 --ope
```

Service appliance set consists of service appliances (Either physical device (F5) or Virtual machine) for loadbalancing the traffic.

```
/opt/contrail/utils/service_appliance.py --api_server_ip 10.0.170.30 --api_server_port 8082 --oper ad
```

Note: tcp cloud OpenContrail packages and OpenContrail lauchpad have service_applice.py scripts in /usr/lib/

Finally there must be created vipnet with subnet propagated on F5 interface. This subnet must be created for vip allocation.

CREATING LOAD BALANCER VIA NEUTRON LBAAS

We booted two instances with apache web server on port 80 into 172.16.50.0/24. This network is terminated in LB VRF. Use the following steps to create a load balancer in Contrail.

Create a pool for HTTP:

```
neutron lb-pool-create --lb-method ROUND_ROBIN --name mypool --protocol HTTP --subnet-id 99ef11f3-a04
```

Add members into the pool:

```
neutron lb-member-create --address 172.16.50.3 --protocol-port 80 mypool
neutron lb-member-create --address 172.16.50.4 --protocol-port 80 mypool
```

Create and associate VIP into the pool. After this command the F5 configuration is applied:

```
neutron lb-vip-create --name myvip --protocol-port 80 --protocol HTTP --subnet-id vipsubnet mypool
```

Finally, create a sample health monitor:

```
neutron lb-healthmonitor-create --delay 20 --timeout 10 --max-retries 3 --type HTTP
```

Associate a health monitor to a pool:

```
neutron lb-healthmonitor-associate <healthmonitor-uuid> mypool
```

When you login into F5 management dashboard, you have to switch into a new partition, which is dynamically created with each LbaaS instance.

Partition: uuid_9b9aa826-5bee-4332-ae96-ac6ce32cf55e   Log out

Local Traffic -> Network Map shows map all objects created and configured by F5 driver.

Green point shows that everything is available and active. If you select Virtual Servers, there is detail of created VIP.

General Properties

Name	uuid_b669d209-30cb-4012-af1e-a337728e8aa3
Partition / Path	uuid_9b9aa826-5bee-4332-ae96-ac6ce32cf55e
Description	myvip:
Type	Standard
Source Address	0.0.0.0/0
Destination Address	185.22.97.18
Service Port	80 HTTP
Notify Status to Virtual Address	☑
Link	None
Availability	● Available (Enabled) - The virtual server is available
Syncookie Status	Off
State	Enabled

Last screenshot captured selected VLAN for VIP.
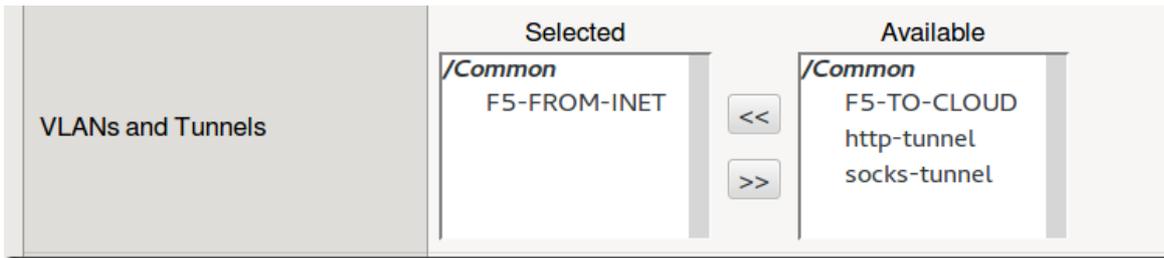
HEAT ORCHESTRATION

As already mentioned at begging the goal is to manage F5 same like other OpenStack resources thru Heat engine. To enable heat orchestration for LbaaS with F5, there must be resource for neutron lbaas provider, which was added in OpenStack Liberty. Therefore we had to backported this resource into OpenStack Juno and Kilo. This link contains gerrit review for lbaas provider https://review.openstack.org/#/c/185197/

Note: *You can use our Ubuntu repositories, where this feature is included*
http://www.opentcpcloud.org/en/documentation/packages-and-repositories/

We prepared sample template for f5 lbaas provider, which can be downloaded and customized as required.
https://github.com/tcpcloud/heat-templates/blob/master/templates/lbaas_contrail_f5_test.hot
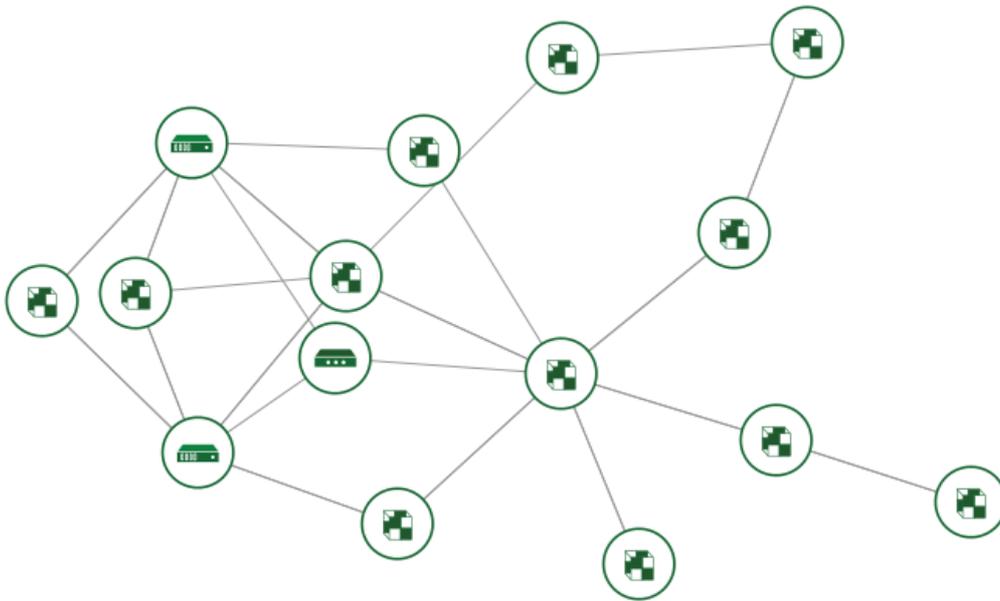
When we have a template with appropriate parameters we can launch stack:

```
heat stack-create -e env/test_contrail_f5_lbaas/demo_ce.env -f template/test_contrail_f5_lbaas.hot te
```

Check the status:

```
heat stack-list

+------------------------------------+-------------------------------+-----------------+---------
| id                                 | stack_name                    | stack_status    | creation_time     |
+------------------------------------+-------------------------------+-----------------+---------
| a4825267-7444-46af-87da-f081c5405470 | test_contrail_f5_lbaas_demo_ce | CREATE_COMPLETE | 2015-10-0
+------------------------------------+-------------------------------+-----------------+---------
```

Describe resources in this stack and verify balancer configuration.

```
xxx:/srv/heat/env# heat stack-show test_contrail_f5_lbaas_demo_ce
+----------------------+--------------------------------------------
| Property | Value
+----------------------+--------------------------------------------
| capabilities    |[]| creation_time | 2015-10-02T12:18:06Z
| description    | Contrail F5 LBaaS Heat Template
| id    | a4825267-7444-46af-87da-f081c5405470
| links    | http://10.0.170.10:8004/v1/2c114f (self)|| notification_topics |[]| outputs |[]
| parameters    |{||"OS::project_id": "2c114f0779ac4367a94679cad918fbd4",
||"OS::stack_name": "test_contrail_f5_lbaas_demo_ce",
||"private_net_cidr": "172.10.10.0/24",
||"public_net_name": "public-net",
||"key_name": "public-key-demo",
||"lb_name": "test-lb",
||"public_net_pool_start": "185.22.120.100",
||"instance_image": "ubuntu-14-04-x64-1441380609",
||"instance_flavor": "m1.medium",
||"OS::stack_id": "a4825267-7444-46af-87da-f081c5405470",
||"private_net_pool_end": "172.10.10.200",
||"private_net_name": "private-net",
||"public_net_id": "621fdf52-e428-42e4-bd61-98db21042f54",
||"private_net_pool_start": "172.10.10.100",
||"public_net_pool_end": "185.22.120.200",
||"lb_provider": "f5",
||"public_net_cidr": "185.22.120.0/24",
||||}| parent    | None
| stack_name    | test_contrail_f5_lbaas_demo_ce
| stack_owner    | demo
| stack_status    | CREATE_COMPLETE
| stack_status_reason  | Stack CREATE completed successfully
| stack_user_project_id | 76ea6c88fdd14410987b8cc984314bb8
| template_description  | Contrail F5 LBaaS Heat Template
| timeout_mins    | None
| updated_time    | None
+----------------------+--------------------------------------------------------------
```

This template is sample, so you have to manually configure Route Target for private net or try to use Contrail heat resources, which is not part of this blog post.

CONCLUSION

We demonstrated that OpenContrail is the only one SDN solution, which enables to manage physical F5 through Neutron LbaaS API instead of own management portal. The next step is implementation of this feature at our pilot customers, where we want to continue on production testing scenarios. Future release should also provide dynamic MX configuration, multi-tenancy, etc.

Jakub Pavlik & Marek Celoud
TCP Cloud Engineers