Load Balancing as a Service v2.0

Liberty and Beyond

Brandon Logan

IRC: blogan

Franklin Naval

IRC: fnaval

Michael Johnson

IRC: johnsom

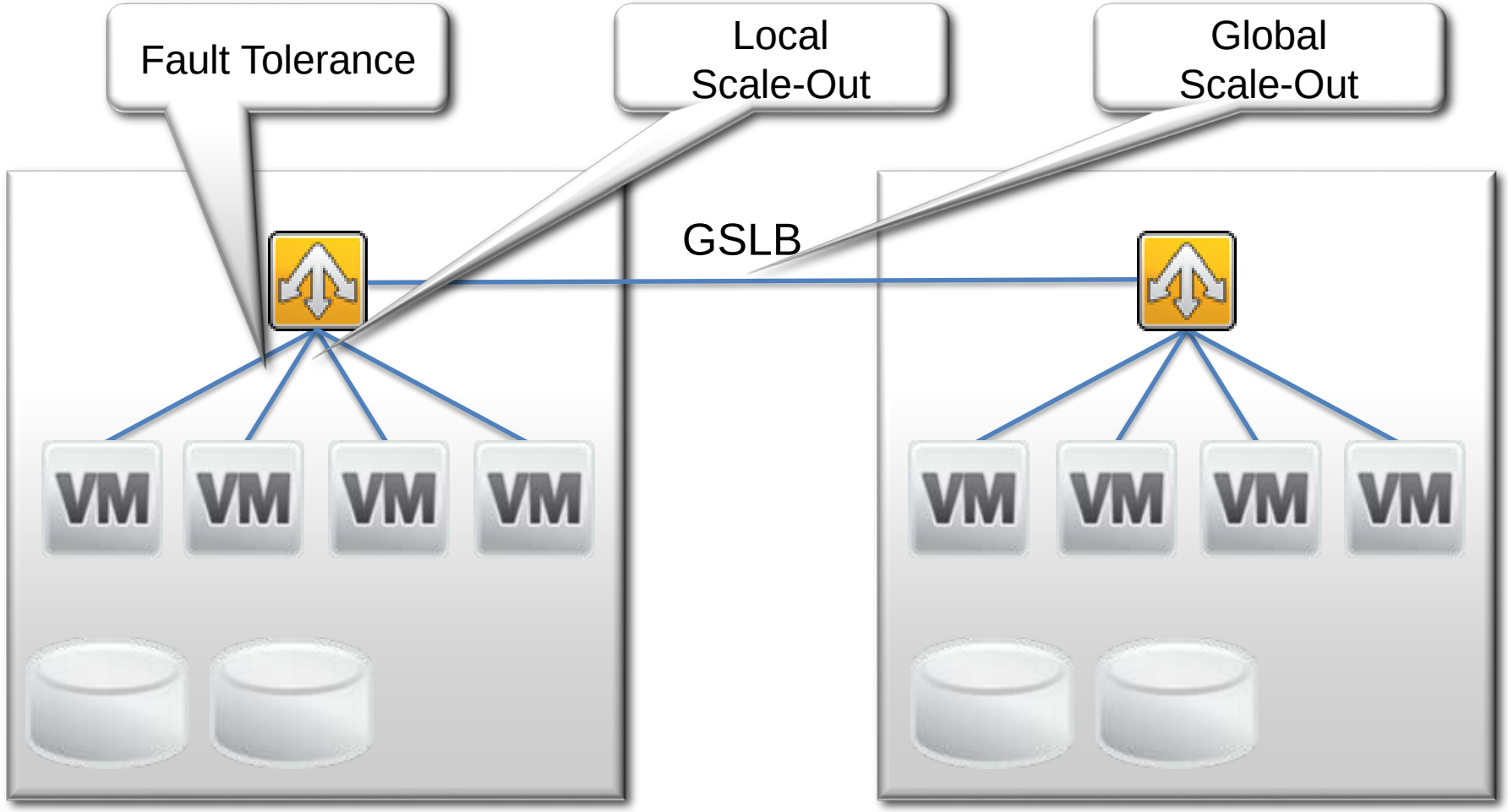
Stephen Balukoff

IRC: sbalukoff

Agenda

- LBaaS Overview (sbalukoff)
  - V1 Overview
- LBaaS v2.0 in Liberty (blogan)
  - Horizon Dashboard
- Testing (fnaval)
- LBaaS v2.0 in Mitaka (blogan)
  - L7, Pool Sharing, Single Create LB
- Octavia (johnsom)
  - Overview
  - Demo - Active/Standby
- Kosmos (johnsom)
- Q & A / Panel discussion

Why is LBaaS critical for cloud applications?



Who's Involved?

LBaaS v1.0 Overview

What was available in LBaaS v1

- L4 load balancing for HTTP, HTTPS pass-through and TCP
- Persistency, including cookie based
- Cookie insertion
- Driver interface for 3rd party products

Problems with LBaaS v1

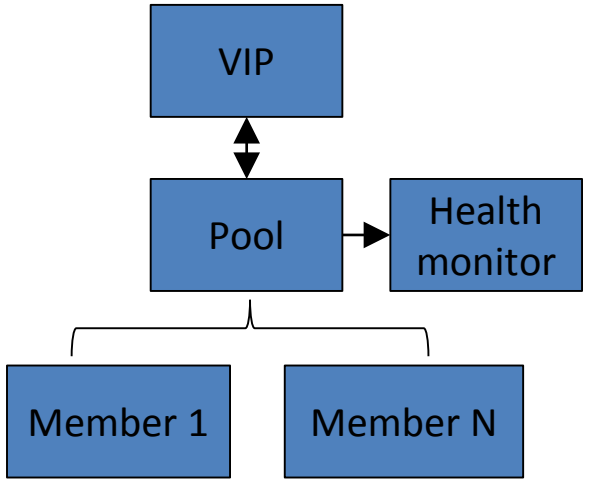
It’s all about the model!

- Not following industry standards (terminology or concepts)
- Barely able to deliver basic “industry” feature set
- Feature improvements are difficult hacks
- Reference driver not scalable
- Nothing scalable without working around model and standard project features (scheduler, etc.)
- Tenant API was dead-end polluting user mindspace
- No advanced Cloud Operator controls

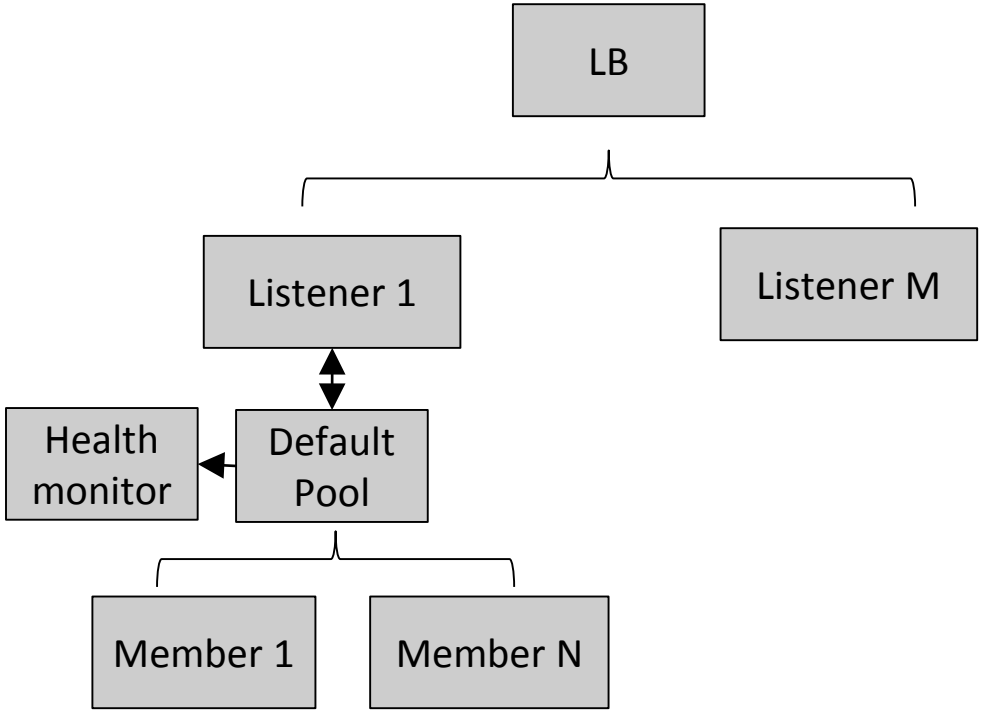
LBaaS v2.0

LBaaS v1 vs. v2

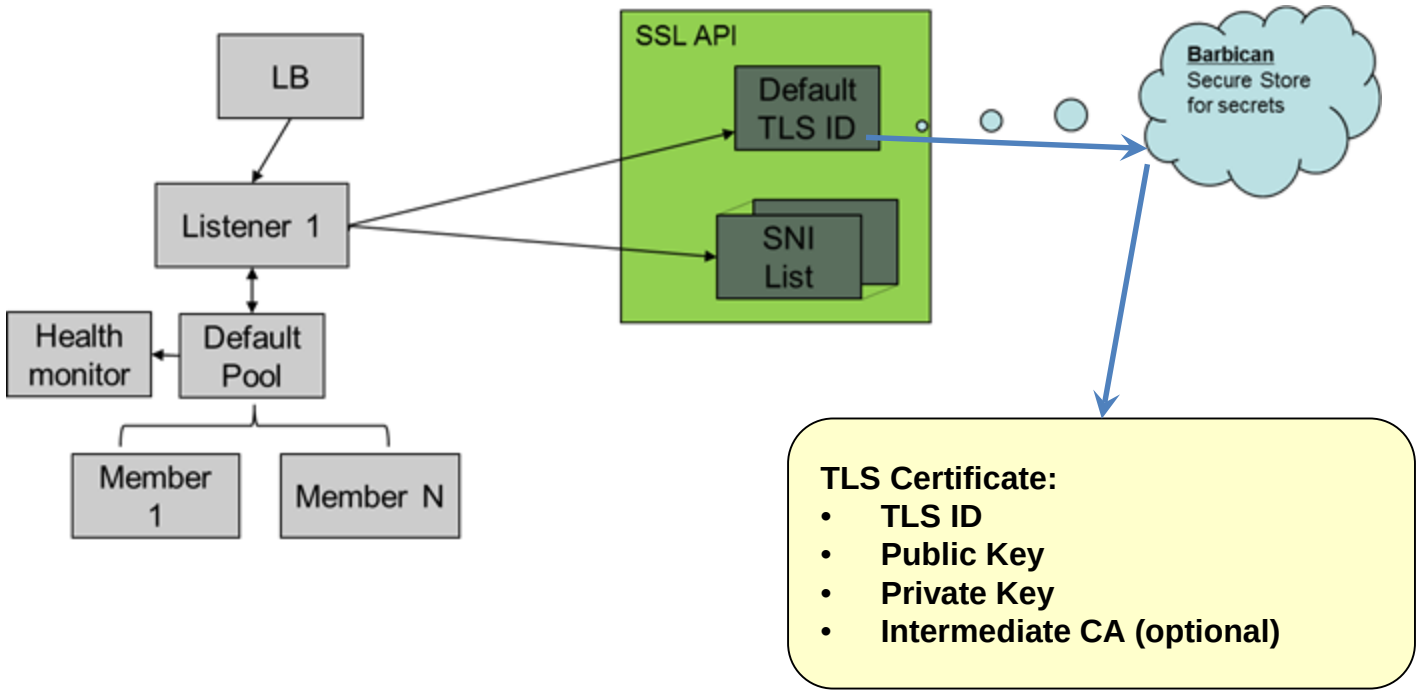
LBaaS v1



LBaaS v2



LBaaS v2 - TLS


LB
Listener 1
Health monitor
Default Pool
Member 1
Member N
SSL API
Default TLS ID
SNI List
Barbican
Secure Store for secrets
TLS Certificate:
• TLS ID
• Public Key
• Private Key
• Intermediate CA (optional)

Community Drivers

LBaaS v1

- Haproxy (lbaas-agent)
- A10 Networks
- Radware
- Citrix
- Embrane
- Vmware

- v1 drivers are NOT supported in v2

LBaaS v2

- Haproxy (lbaas-agent)
- Octavia
- A10 Networks
- Radware
- Brocade
- Citrix
- KEMP

LBaaS v2.0 in Liberty

Progress in Liberty

- Neutron LBaaS V2 out of experimental
- Neutron LBaaS V1 deprecated
- Octavia as the Reference Implementation
- Large chunk of work done on the V2 Horizon Dashboard
- Large chunk of work done for L7 Content Switching

Testing

- Initially, Kilo only had unit tests and a few tempest tests.
- For Liberty, there was a concerted effort to improve the test coverage.
  - Functional tempest tests
  - Data driven tests
  - Scenario tempest tests

Functional Tests

- Developed clients that interact with the API
- 100% positive API test coverage
- Substantial negative tests added

Data Driven Tests

- Many different configuration permutations
- DDT easily iterates through those permutations each as its own test
- Uncovered many bugs

Data Driven Tests

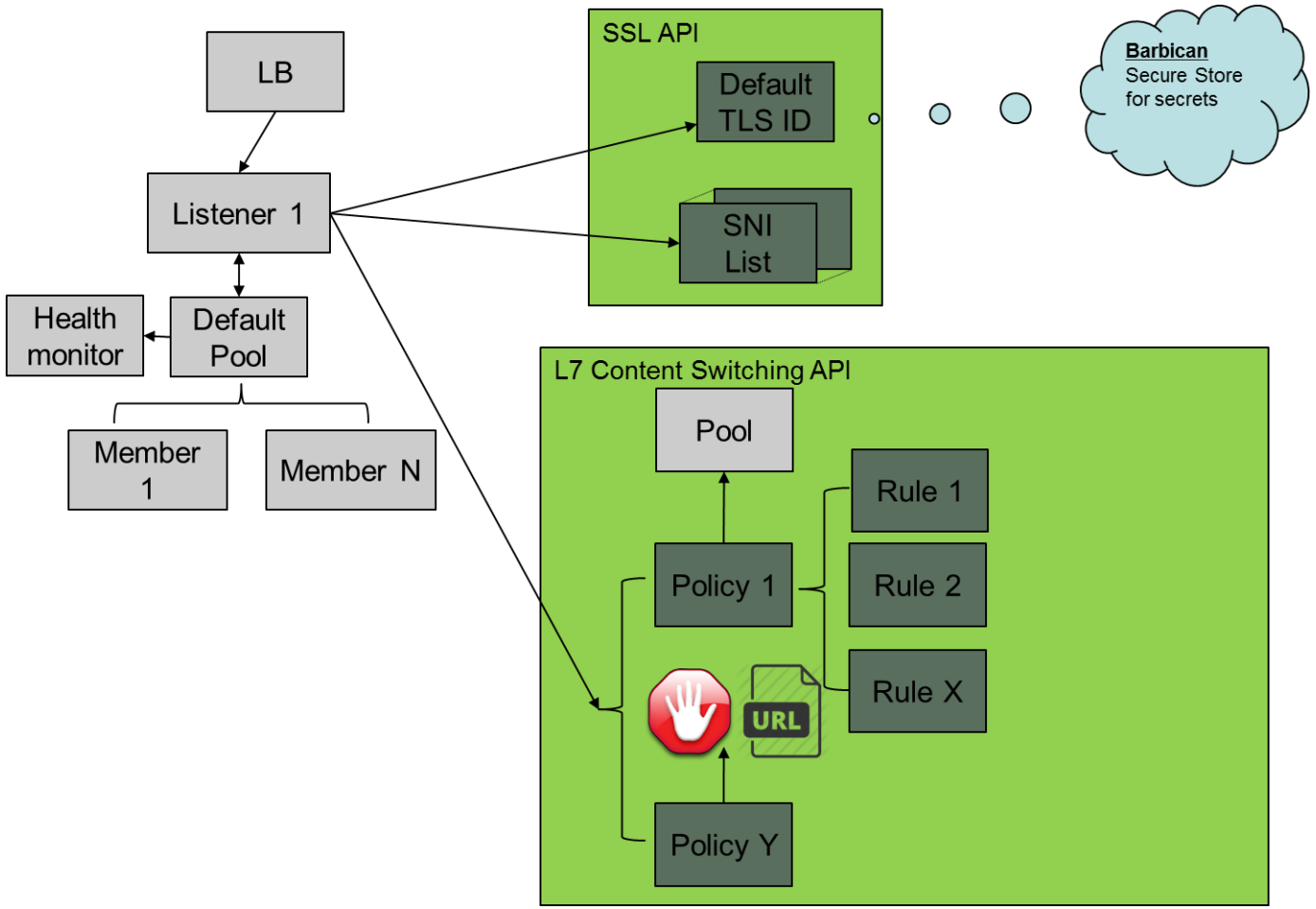
```
from oslo_log import log as logging
import testscenarios

from neutron_lbaas.tests.tempest.lib import config
from neutron_lbaas.tests.tempest.v2.ddt import base_ddt

CONF = config.CONF

LOG = logging.getLogger(__name__)

"""
Tests the following operations in the Neutron-LBaaS API using the
REST client for Listeners:

     |------|---------------------|---------------------|-----------------------------------|
     |S.No  |Action               |LB admin_state_up    | Listener admin_state_up           |
     |------|---------------------|---------------------|-----------------------------------|
     | 1    | Create Listener     | True                | True                              |
     | 2    |                     | True                | False                             |
     | 3    |                     | False               | True                              |
     | 4    |                     | False               | False                             |
     | 5    | Update Listener     | True                | True  --> True                    |
     | 6    |                     | True                | True  --> False                   |
     | 7    |                     | True                | False --> True                    |
     | 8    |                     | True                | False --> False                   |
     | 9    |                     | False               | True  --> True                    |
     | 10   |                     | False               | True  --> False                   |
     | 11   |                     | False               | False --> True                    |
     | 12   |                     | False               | False --> False                   |
     |------|---------------------|---------------------|-----------------------------------|

"""
# set up the scenarios
scenario_lb_T = ('lb_T', {'lb_flag': True})
scenario_lb_F = ('lb_F', {'lb_flag': False})

scenario_listener_T = ('listener_T', {'listener_flag': True})
scenario_listener_F = ('listener_F', {'listener_flag': False})

scenario_lis_to_flag_T = ('listener_to_flag_T', {'listener_to_flag': True})
scenario_lis_to_flag_F = ('listener_to_flag_F', {'listener_to_flag': False})

# The following command creates 4 unique scenarios
scenario_create_member = testscenarios.multiply_scenarios(
        [scenario_lb_T, scenario_lb_F],
        [scenario_listener_T, scenario_listener_F])

# The following command creates 8 unique scenarios
scenario_update_member = testscenarios.multiply_scenarios(
    [scenario_lis_to_flag_T, scenario_lis_to_flag_F],
    scenario_create_member)
```

Scenario Tests

- Unit and functional tests do not test end to end
- Need tests that verify packets flow as expected
- Also verifies communication between dependencies are working as intended

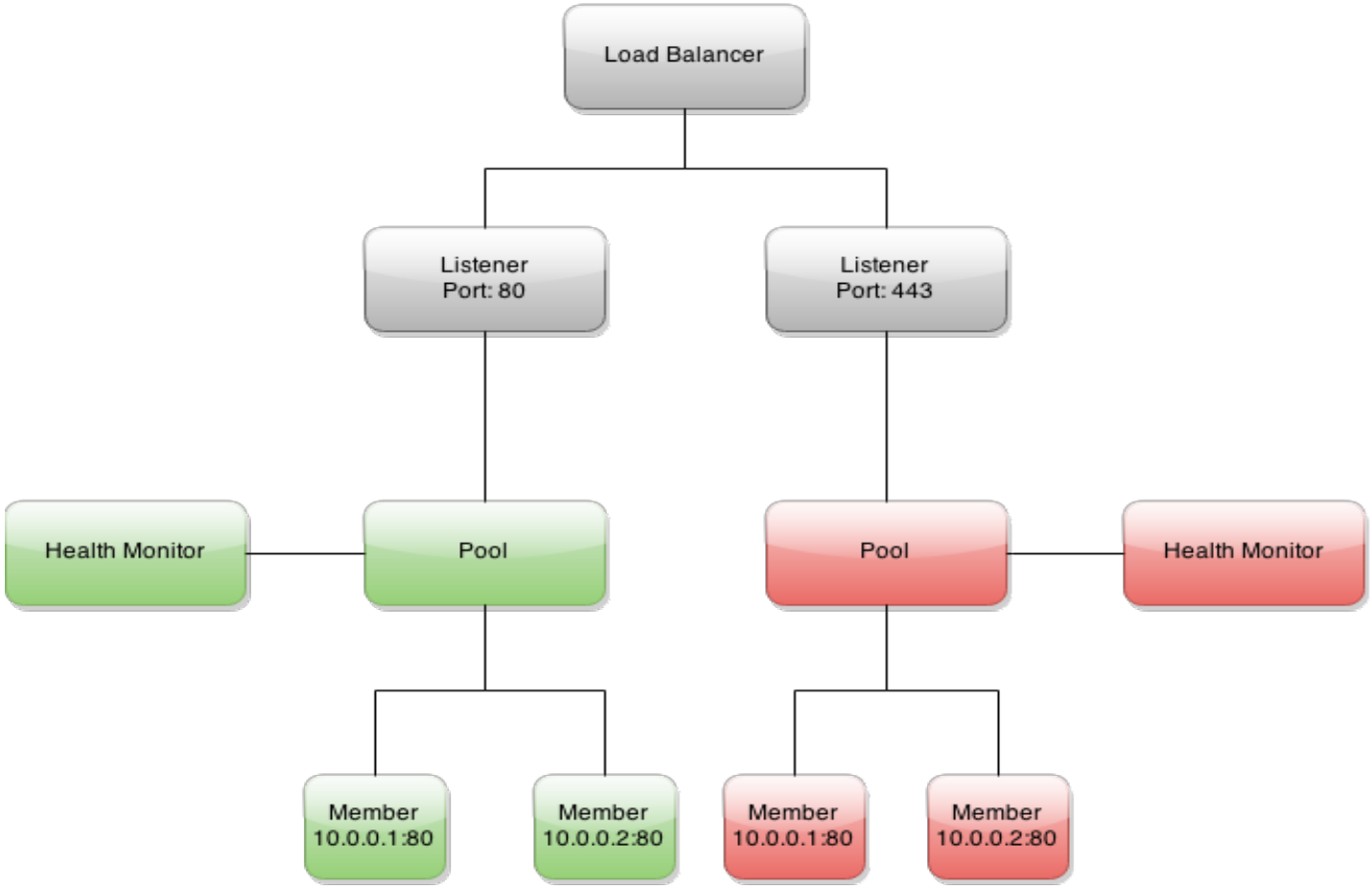
LBaaS v2.0 in Mitaka

L7 Content Switching

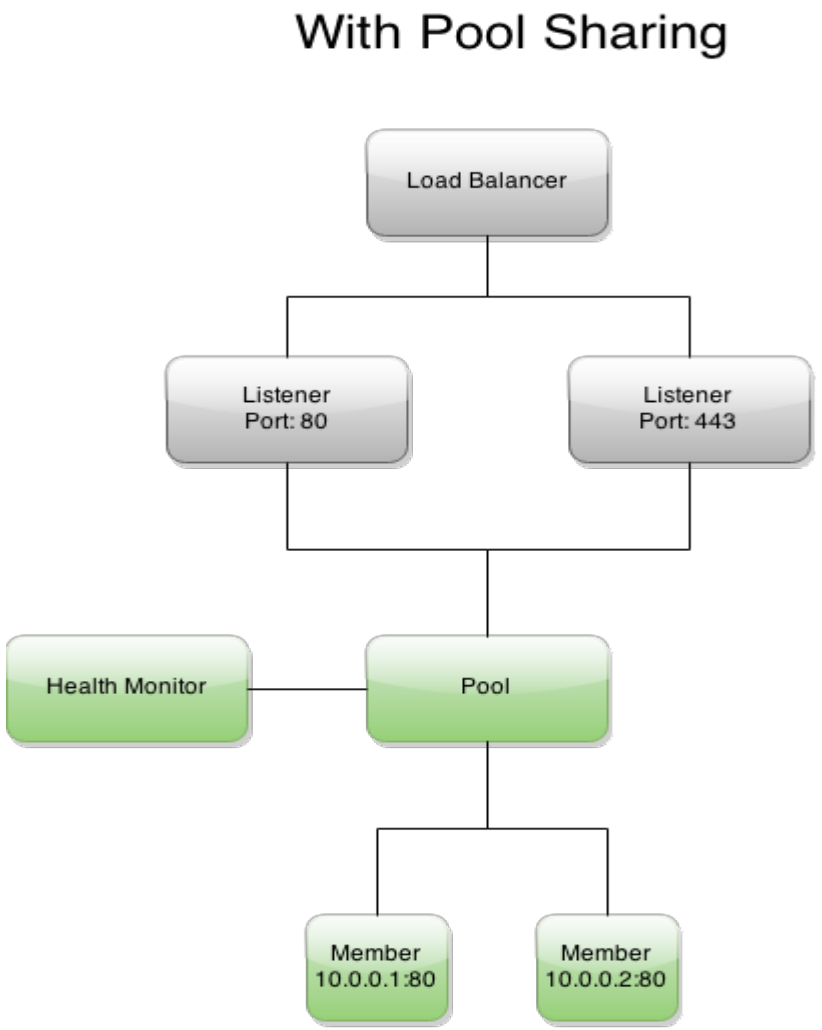


Pool Sharing

Without Pool Sharing



Pool Sharing



Single API Request

Currently :

1. API request to create load balancer
2. API request to create listener
3. API request to create pool
4. API request to create member

Single Create API Request

1. Provide entire load balancer tree in a single API request
    - Only one API Request needed
    - Entire configuration provided to drivers up front
    - Easier for horizon

Flavor Framework For Neutron Advanced Services

Flavor is a named resource used to schedule a provider driver with metadata at resource creation.

Example flow:

- Operator:
  - Creates named flavors “Gold”, “Silver” and “Bronze” for service type LOADBALANCERV2
  - Creates *service profiles* that represent desired provider drivers and metadata
  - Associates desired service profiles with flavors
    - Example: associate the “Gold” flavor with a service profile for the Octavia Active/Standby
- User:
  - Specifies desired flavor (ex: “Gold”) as parameter on resource creation
    - Example: create load balancer
  - The flavor is used to pick a currently relevant provider and creates the resource
    - Example: Octavia Active/Standby

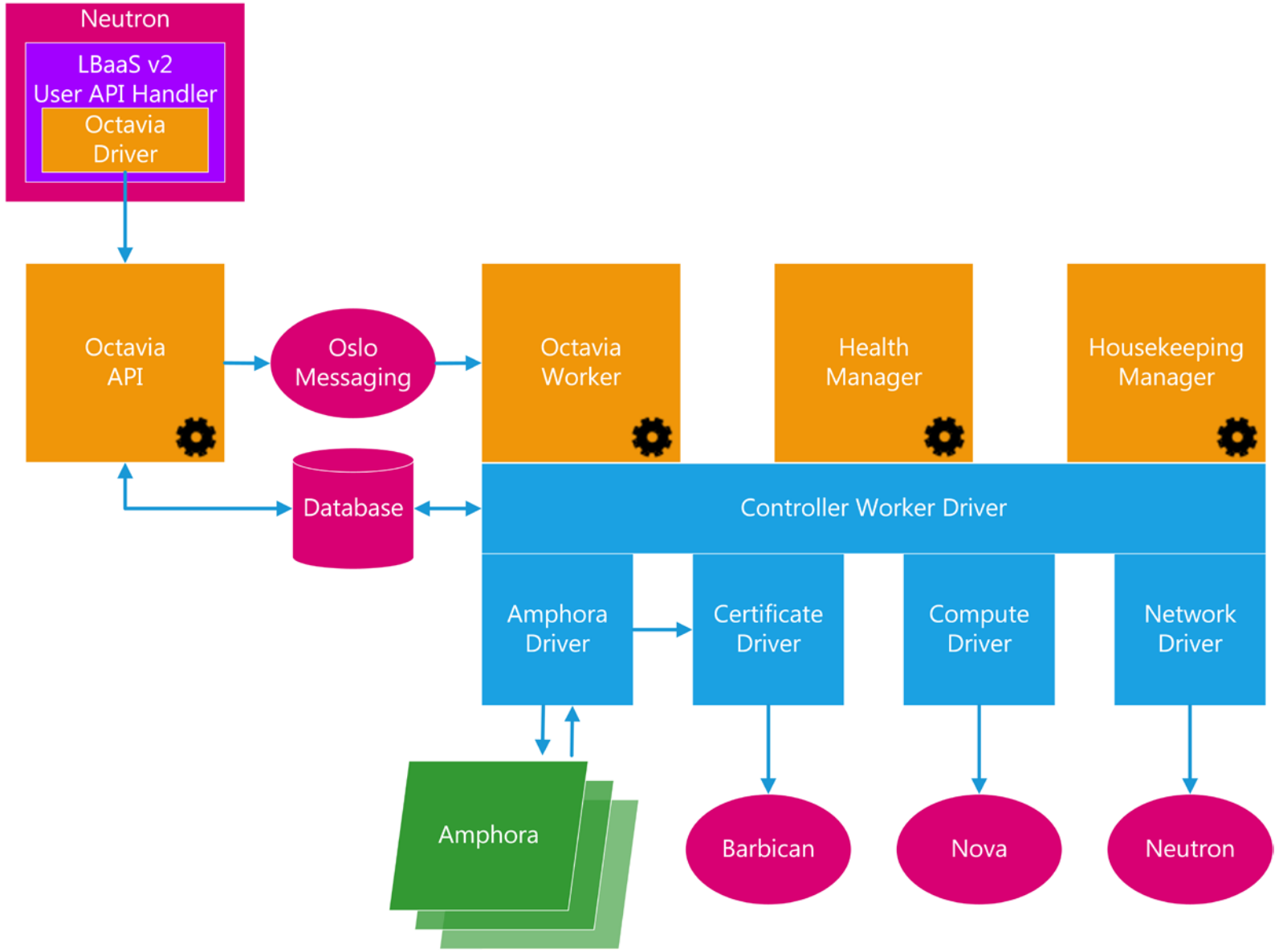
Gives operator dynamic control of providers and metadata used for resource creation.

- Associate “Silver” with the Octavia driver using metadata indicating hot spare failover
- When a user creates a load balancer using a “Silver” flavor, the Octavia driver with hot spare failover

Octavia

OpenStack Load Balancing

Octavia Component Design v0.5



Octavia Roadmap

Note: This roadmap WILL change based on the design sessions this week.

Octavia v0.5 Liberty ✓

- Feature parity with existing reference driver
- Service virtual machines
- Spares pool failover

Octavia v1.0 – Mitaka?

- Amphora Active/Standby
- High Availability control plane
- Layer 7 rules
- Container support
- Flavor framework support

Octavia v2.0?

- Active/Active amphora
- Amphora horizontal scale

On to the demo!

Try Octavia yourself on DevStack

In your localrc add:

enable_plugin neutron-lbaas https://git.openstack.org/openstack/neutron-lbaas

enable_plugin octavia https://git.openstack.org/openstack/octavia.git

ENABLED_SERVICES+=,q-lbaasv2,octavia,o-cw,o-hk,o-hm,o-api

Operator API is at: http://127.0.0.1:9876

Operator API documentation: http://www.octavia.io/review/master/main/octaviaapi.html

neutron client: neutron lbaas-[loadbalancer-create]

Sample Vagrant and local.conf files are available under octavia/devstack/samples

OpenStack Octavia

We are looking for contributors!

For more information:

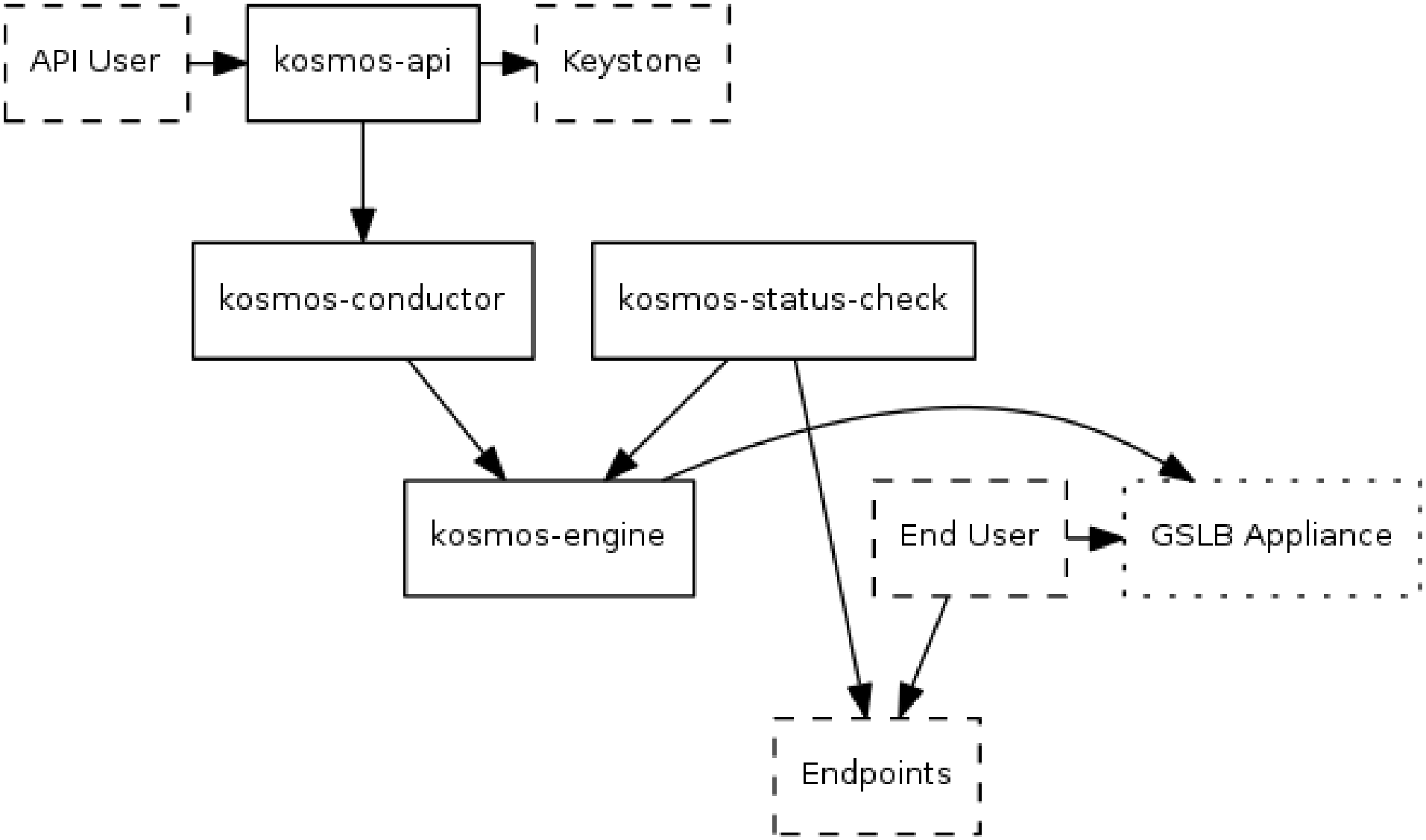
Freenode IRC: #openstack-lbaas

Weekly meetings: Wednesdays at 20:00 UTC on #openstack-meeting-alt

- https://wiki.openstack.org/wiki/octavia
- http://www.octavia.io
- https://launchpad.net/octavia
- https://github.com/openstack/octavia


KOSMOS
OPENSTACK GLOBAL LOAD BALANCING

Kosmos System Overview



OpenStack Kosmos

We are looking for contributors!

For more information:

- https://wiki.openstack.org/wiki/Kosmos
- https://launchpad.net/kosmos
- https://github.com/openstack/kosmos

Freenode IRC: #openstack-gslb

Weekly meetings: Tuesdays at 1600 UTC on #openstack-meeting-4

PTL: Graham Hayes (irc: mugsie)

Cores: Doug Wiegley (irc: dougwig)

Michael Johnson (irc: johnsom)

Craig Letourneau - CC0 1.0 Universal

Q & A / Panel discussion

https://wiki.openstack.org/wiki/Neutron/LBaaS
https://wiki.openstack.org/wiki/Octavia
https://wiki.openstack.org/wiki/Kosmos

IRC: #openstack-lbaas

We are: IRC:sbalukoff, IRC:blogan, IRC:fnaval, IRC:johnsom