OpenStack Meetings || <https://wiki.openstack.org/wiki/Meetings>]

[21:03] <joehuang> hi

[21:03] <zhiyuan> hi

[21:09] <joehuang> hi gampel

[21:10] <gampel> Hi sorry for the delay

[21:10] <joehuang> it's ok

[21:10] <joehuang> let's start the meeting

[21:10] <gampel> ok

[21:11] <joehuang> #startmeeting tricircle

[21:11] <@openstack> Meeting started Wed Jul 15 13:11:55 2015 UTC and is due to finish in 60 minutes.  The chair is joehuang. Information about MeetBot at <http://wiki.debian.org/MeetBot>.

[21:11] <@openstack> Useful Commands: #action #agreed #help #info #idea #link #topic #startvote.

[21:12] == openstack changed the topic of #openstack-meeting to:  (Meeting topic: tricircle)

[21:12] <@openstack> The meeting name has been set to 'tricircle'

[21:12] <joehuang> #topic rollcall

[21:12] == openstack changed the topic of #openstack-meeting to: rollcall (Meeting topic: tricircle)

[21:12] <joehuang> #info joehuang

[21:12] <gampel> #info gampel

[21:12] <zhiyuan> #info zhiyuan

[21:13] <joehuang> #topic design discussion

[21:13] == openstack changed the topic of #openstack-meeting to: design discussion (Meeting topic: tricircle)

[21:13] <irenab> #info irenab

[21:13] <joehuang> #info the cascade service should be able run with multiple instances

[21:14] <gampel> yes i agree

[21:14] <joehuang> therefore, I think it should work like Nova-scheduler

[21:14] <gampel> One option that we've already discussed to avoid contention is that each service is responsible for different bottom sites.

[21:14] <joehuang> this is one option

[21:15] <joehuang> anathor option is that cascade service work like nova-scheduler

[21:15] <joehuang> multiple Nova-scheduler will work in parallel

[21:15] <joehuang> but only one will be selected for a boot vm request

[21:16] <gampel> but then you will have only one cascade service

[21:16] <irenab> joehuang: this should cover both load balancing and hight availability for the cascade service, right?

[21:16] <gampel> responsible  for the scheduling

[21:16] <joehuang> No, multiple nova-scheduler can work together

[21:17] <joehuang> but I did not look at the code how it works

[21:17] <gampel> But it is not only nova lets say you add port to a router

[21:18] <gampel> If I understand  correctly you mean to add it in the Nova and Neutron core plugin

[21:18] <joehuang> the cascade service to send request to multiple bottom OpenStack

[21:18] <joehuang> the cascade service to send request to multiple bottom OpenStack if needed

[21:18] <gampel> I am not sure i understand  where you want to do the sch before the MQ or after

[21:19] <joehuang> not shedule, but like one queue, multiple cascade consume the same queue

[21:19] <joehuang> whenever one cascade pickup the message from the queue

[21:20] <joehuang> the message will be removed from the queue

[21:20] <zhiyuan> i think joe's idea is that each cascade service is aware of all the bottom openstack

[21:20] <joehuang> then only one cascade service will process the message

[21:20] <joehuang> to zhiyuan, currently in the design doc, we are thinking in this way

[21:20] <zhiyuan> every cascade service can service the request independently, am I correct?

[21:21] <joehuang> all bottom information could be seen from the database

[21:21] <joehuang> yes

[21:22] <joehuang> zhiyuan, you mentioned that if the message is sending not using fanout, then it work like this way

[21:23] <joehuang> to irenab: it's not loadbalancing, how we use the message bus

[21:24] <gampel> So you want to use queue, in the POC where there any sequence of API requests that were related ?

[21:24] <zhiyuan> yes, services consume the same queue with the same topic if fanout is not used

[21:24] <irenab> joehuang: thank you for clarification

[21:25] <joehuang> So I just give an example, that the way for nova-scheduler using the message bus is what we can leverage

[21:25] <gampel> So all the services will be authenticated  to all the bottom OpenStack

[21:26] <joehuang> using the token carried in the top API request

[21:26] <gampel> You mean the token will be in the DB ?

[21:27] <joehuang> same to token to the bottom OpenStack instance, it's carried in the contex. not in DB

[21:27] <gampel> or just proxy

[21:27] <joehuang> same token to the bottom OpenStack instance, it's carried in the contex. not in DB

[21:28] <gampel> Did it work in the POC  using same token on all the bottom OpenStacks

[21:28] <joehuang> not only proxy, cascade serice will do long-run task

[21:28] <gampel> which long-run task ?

[21:29] <zhiyuan> POC uses one single Keystone, so I think using the same token is fine

[21:29] <joehuang> it works in the PoC. for background periodic task, some configured user is used to access the bottom OpenStack

[21:30] <joehuang> Federated KeyStone not tried in the poc

[21:30] <gampel> which background periodic task ?

[21:30] <joehuang> periodic to poll the VM/volume/port status

[21:31] <joehuang> so that the status in the top layer will be refreshed

[21:31] <joehuang> but if we have interception at the API layer, we can change this way

[21:31] <gampel> At the moment as a start point we said that we going to pass run time info to the bottom layer

[21:32] <joehuang> the status could be queried only when API request coming, and return stale status first, and refrest the status immedidtly for next time query

[21:33] <joehuang> to gampel, what you mean "pass run time info to the bottom layer"

[21:34] <joehuang> through the delay refresh for next time, we can reduce to use a configured user info to sync the status from the bottom openstack

[21:34] <gampel> I mean if you get a port status from the API you pass it to the cascading service that will query the bottom Opensatck

[21:35] <gampel> But  if is is configuration request  query get it from the DB

[21:35] <joehuang> Ok, but if there is a lot of ports, the latency for two hop query will be longer

[21:37] <gampel> You assume the query request will arrive together, is that what happened in the POC

[21:37] <joehuang> and if multiple bottom openstack involved, it's complex to query at the same time

[21:38] <joehuang> a query can be issued with different filter, and the ports meet the filter will be returned

[21:38] <joehuang> the filter can be varied a lot

[21:39] <gampel> Ok l suggest  to look at it know as an optimization that we can add latter after we have the basic staff working

[21:40] <joehuang> ok

[21:41] <gampel> I want to update regarding the status of the experimental work status

[21:41] <joehuang> I think the current design is a very good beginning to start

[21:41] <joehuang> so how about nova

[21:41] <gampel> we have the infrastructure for the foundation service

[21:42] <gampel>  neutron core plugin on top pass the request on the message queue up to the service endpoint

[21:42] <joehuang> I mentioned in the mail we need to inherited from the API class, just like cells does

[21:42] <gampel> nova is almost done and will be checked in on Sunday

[21:43] <gampel> yes we will send it soon

[21:43] <joehuang> ooo, that;s great

[21:43] <zhiyuan> cool

[21:43] <joehuang> I think once nova and neutron work, we can move it to the master branch

[21:43] <gampel> if you like we have the infrastructure now to work together on it

[21:44] <joehuang> so that we can also to add code on it

[21:44] <gampel> next step is to create  the DB alchemy module to the cascading service

[21:45] <gampel> devstack is ready and working as well for neutron request

[21:45] <joehuang> zhiyuan, how about to study how to make multiple cascade service work in parallel: multi-worker, multi-instances

[21:45] <joehuang> I think it's necessary

[21:46] <joehuang> gampel, could you upload code to git for review

[21:46] <gampel> we will update the patches and merge them all into the experimental branch on Sunday

[21:46] <gampel> so you could join the work next week

[21:47] <joehuang> that's super cool, then we can work together on the new staff

[21:47] <zhiyuan> no problem, i can try based on saggi's patch

[21:47] <gampel> there are now 3 patches by saggi covering neutron devstack and the cascading service including the endpoint

[21:48] <gampel> probably Sunday we will have Nova in as well

[21:48] <zhiyuan> and I will set up the new CI job after all the new code is submitted

[21:48] <joehuang> perfect, zhiyuan

[21:48] <gampel> can we add test to the CI to do pep8 at least

[21:49] <joehuang> let's try it

[21:49] <zhiyuan> sure

[21:49] <gampel> thx

[21:50] <joehuang> how about the topic in Tokyo summit

[21:51] <gampel> i summited today with irena  me  and joe as speakers

[21:51] <joehuang> CJK project will have a poc of cascading/tricircle,

[21:52] <gampel> can you elaborate about what is CJK ?

[21:52] <joehuang>  CJK submitted with one topic: inter-cloud in China, Japan, Korea by Takashi and me and guys from Korea

[21:52] <joehuang> CJK is China, Japan, Korea

[21:53] <joehuang> nice. info sync-ed

[21:53] <gampel> very nice please share the POC result when you have it

[21:54] <joehuang> and OPNFV day, the multisite project is an formal project in OPNFV, so we'll apply one session in OPNFV day in Tokyo summit

[21:54] <joehuang> Sure, we will start soon

[21:55] <joehuang> We have a very good progress

[21:55] <joehuang> let's have a meeting next week

[21:55] <gampel> Ok thx

[21:55] <joehuang> thanks you all

[21:56] <joehuang> see you next time

[21:56] <irenab> thanks you

[21:56] <joehuang> bye

[21:56] <gampel> thank you bye

[21:56] <joehuang> #endmeeting

[21:56] <zhiyuan> see you