Implementations of Billing Functionalities in OpenStack (openstack-bill)

In this work, accountability (i.e. billing) function is added in OpenStack (Diablo release). A secured billing module is integrated in two main projects of OpenStack i.e Keystone which provides identity service and Horizon which provides graphical API service, and to get the details of usage it interacts with Nova which provides compute service

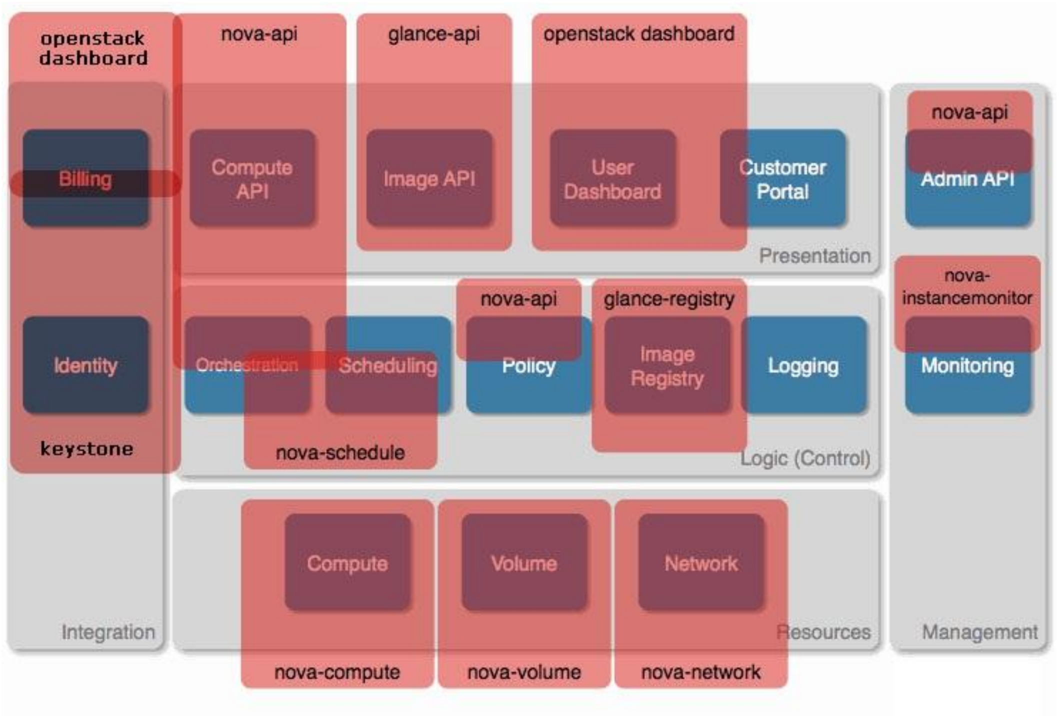


Figure 1 Conceptual Mapping of Implementation

The above Figure show the conceptual coverage of the implementation made in this project. This implementation is made using python programming language version 2.7 as like other project in OpenStack.

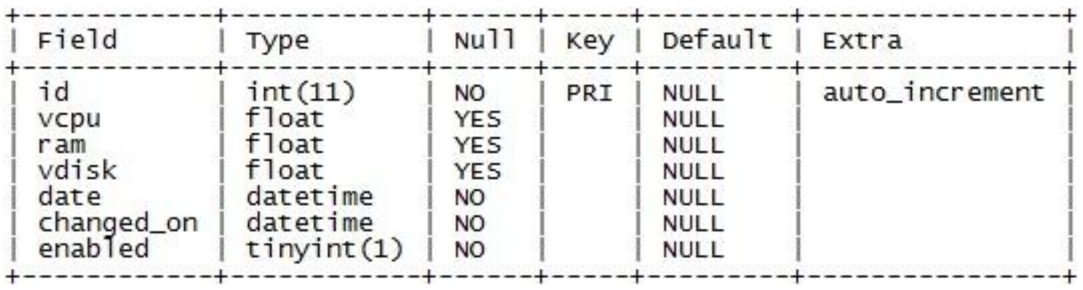
1 Enhanced Keystone

The default version of keystone used by diablo release of OpenStack provides identity service for OpenStack projects. With addition to this default identity service, enhanced keystone will provide the billing service using newly added biller module. A set of new Application program Interface (API) are introduced for providing these services. Three new table named “bill_unit”, “instance_bills” and “user_bills” are created in keystone database to store data regarding to billing system.

1.1 Bill Units

This table is used to store the unit cost of the basic resource provided by Nova i.e. virtual Central Processing Unit (CPU), Random Memory Access (RAM) and virtual Storage like virtual disk. Each entry represents the unit cost of corresponding months. Every month will have a unique entry in the table and are used for calculating the bill for resource usage by the users. These entries can only made by the Admin API. Description of this table is shown in table 1

Table 1 Description of Bill_Unit Table



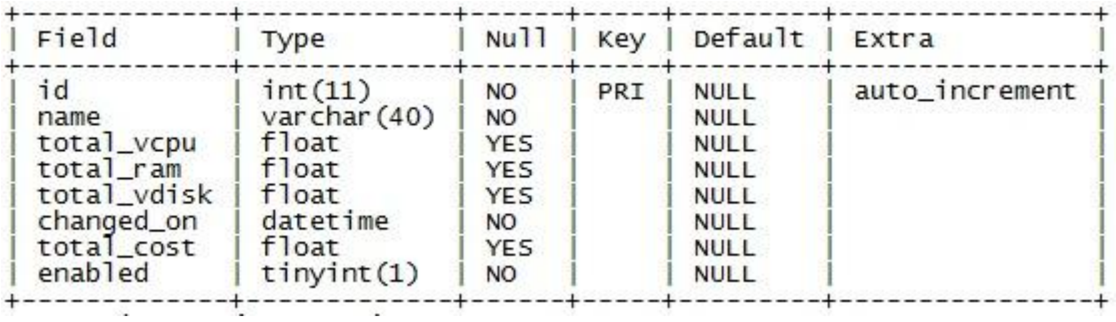
```
+------------+-----------+------+-----+---------+----------------+
| Field      | Type      | Null | Key | Default | Extra          |
+------------+-----------+------+-----+---------+----------------+
| id         | int(11)   | NO   | PRI | NULL    | auto_increment |
| vcpu       | float     | YES  |     | NULL    |                |
| ram        | float     | YES  |     | NULL    |                |
| vdisk      | float     | YES  |     | NULL    |                |
| date       | datetime  | NO   |     | NULL    |                |
| changed_on | datetime  | NO   |     | NULL    |                |
| enabled    | tinyint(1)| NO   |     | NULL    |                |
+------------+-----------+------+-----+---------+----------------+
```

Table bill_unit has seven fields that are used to store the cost of unit measurement of resources. The first field ‘id’ represents the serial number of unit bill entry and it is used as the primary key of the table. Fields like vcpu, ram and vdisk represents cost of a cpu per hour, the cost of 1Mb ram per hour and the cost of 1Gb storage per hour respectively. The calculation and bills are stored as US dollars ($). The month field shows the month for which this unit cost is applicable and changed_on stores the time and date on which this entry is made. Finally ‘enable’ filed show whether the entry is active or not.

1.2 Instance Bills

It stores cost of resources used by the instances so far i.e. bill of instances. The instances table which is maintained in nova project and communicated via nova-api has the instance usage month wise. There are no entries in Nova that stores the instance wise usage despite of number of month it runs. So this table does the job of storing the details about an instance from its start to termination. Description of instance bill is shown in table 2

Table 2 Description of instance_bills Table



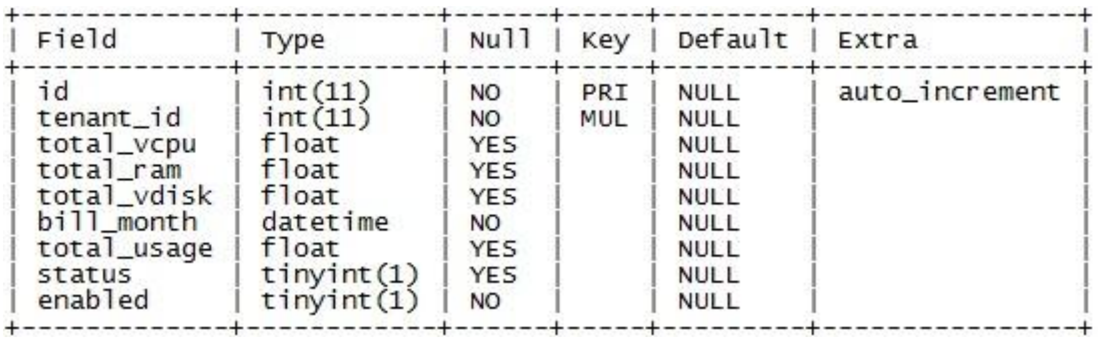
```
+-------------+-------------+------+-----+---------+----------------+
| Field       | Type        | Null | Key | Default | Extra          |
+-------------+-------------+------+-----+---------+----------------+
| id          | int(11)     | NO   | PRI | NULL    | auto_increment |
| name        | varchar(40) | NO   |     | NULL    |                |
| total_vcpu  | float       | YES  |     | NULL    |                |
| total_ram   | float       | YES  |     | NULL    |                |
| total_vdisk | float       | YES  |     | NULL    |                |
| changed_on  | datetime    | NO   |     | NULL    |                |
| total_cost  | float       | YES  |     | NULL    |                |
| enabled     | tinyint(1)  | NO   |     | NULL    |                |
+-------------+-------------+------+-----+---------+----------------+
```

Totally eight fields are used to store the complete details about the running cost of an instance. First field 'id' represents the instance id which is same as the id store in nova database. Enabled field represent the status of an instance i.e. Active or Terminated. And changed_on field says when the table is updated lastly. Remaining fields like name, total_vcpu, total_ram, total_vdisk and total_cost stores the cost of total virtual CPUs usage, total RAM usage, total storage usage and total cost of instance respectively.

1.3 Tenant Bills

Table tenant_bill is used to store the bill of each tenant for every month. Entry is made in this table after the launch of first instance by the user who is registered under corresponding tenant. Since the tenant is viewed as group of user who belongs to same company or project or origination, there is no point is storing the user's bill separately in database. However the tenant can still view the usage and bill of individual user under him. Bills for users are calculated on the basis of instances launched by them. And instances which are launched by different user can be accessed by other users of the same tenant, this show why tenant based billing is considered instead of user based billing. Description of tenant bill is show in table 3.

Table 3 Description of tenant_bills Table



```
+-------------+------------+------+-----+---------+----------------+
| Field       | Type       | Null | Key | Default | Extra          |
+-------------+------------+------+-----+---------+----------------+
| id          | int(11)    | NO   | PRI | NULL    | auto_increment |
| tenant_id   | int(11)    | NO   | MUL | NULL    |                |
| total_vcpu  | float      | YES  |     | NULL    |                |
| total_ram   | float      | YES  |     | NULL    |                |
| total_vdisk | float      | YES  |     | NULL    |                |
| bill_month  | datetime   | NO   |     | NULL    |                |
| total_usage | float      | YES  |     | NULL    |                |
| status      | tinyint(1) | YES  |     | NULL    |                |
| enabled     | tinyint(1) | NO   |     | NULL    |                |
+-------------+------------+------+-----+---------+----------------+
```

The field 'id' represents bill number and 'bill_month' represents the month and year for which the bill is made. Tenant id field in above table is a foreign key whose entry present in the table containing list of tenants. Status and enabled fields stores the details of status of payment made by the user and activeness of the entry respectively. Remaining field storages the usage cost details. Uniqueness in the combination of billing month and tenant is achieved by programming level, because in Django (python web framework) at database level it is not possible to have a composite key between foreign key i.e. tenant id and bill month [57].

These primary database table used for billing modules are coded in such a way that it will work properly no matter what database backend(mysql, memcache, etc) is used in the configuration of keystone (version 2011.3.1). The API which is used for manipulation of contents in theses tables is named as "BILLER" and the controller as "BILLCONTROLLER".

2 Enhanced Horizon

Horizon is OpenStack dashboard implementation which provides web based user interface to OpenStack services. A detailed explanation about this is covered in the previous chapter. Billing functionalities are added with the standard horizon release of OpenStack to more convening for real time deployment in organizational or industrial levels.

Horizon works closely with nova-api and keystone for the billing implementation. It makes use of openstackx (Diablo version) for communication with keystone and nova-api. Openstackx is an API extension for nova-api, keystone services and other services used by OpenStack dashboard. The messages communicated between different projects are formatted by using XML or JSON. The most of the billing calculations and updating are initialized from here. For every month the basic unit cost of resources are created by admin through horizon.

2.1 Creating a unit cost per resources

For billing functionality this step is crucial. Administrator decides the unit cost of resources and login to dashboard to update the values so that all the other billing calculation can be done. Unit cost per resources is the basic unit used for calculating bills for corresponding months. Each month the unit cost can be changed. Figure 3 show sequence of messages during creating a unit bill

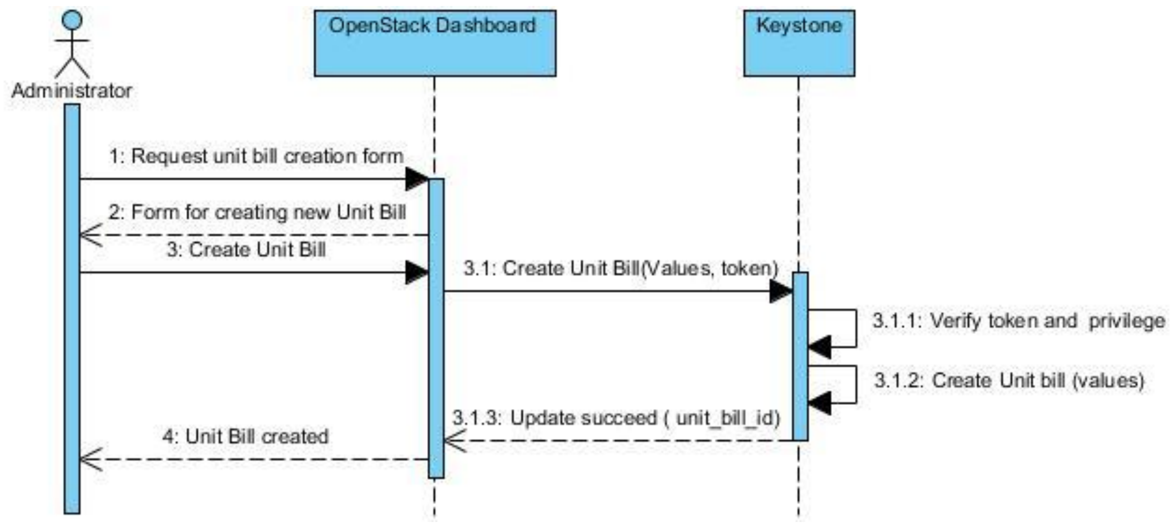
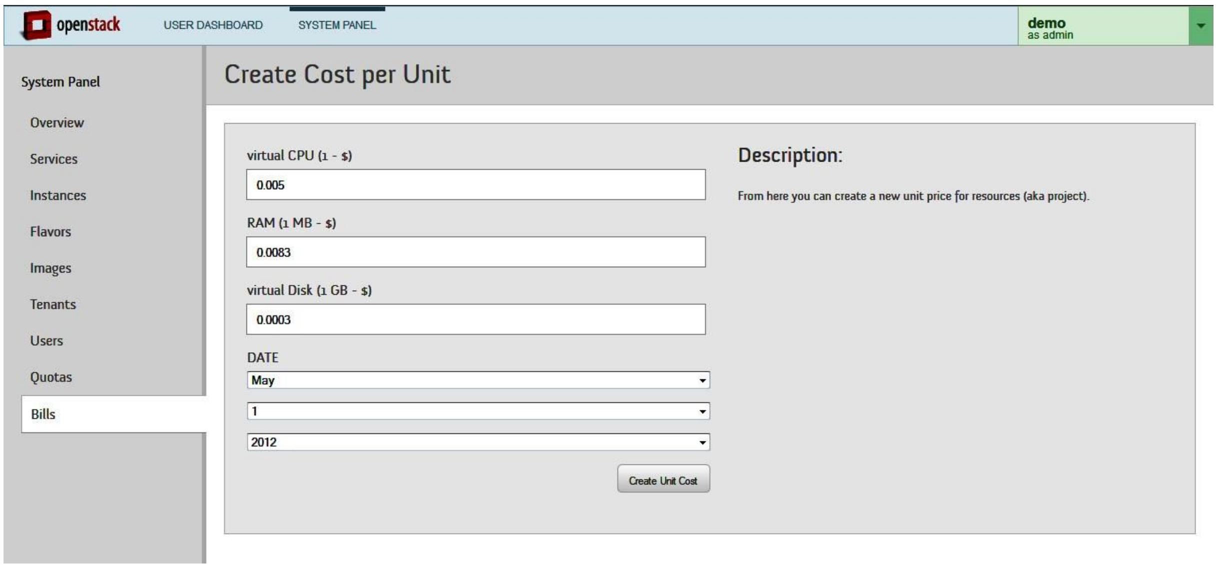

Administrator
OpenStack Dashboard
Keystone
1: Request unit bill creation form
2: Form for creating new Unit Bill
3: Create Unit Bill
3.1: Create Unit Bill(Values, token)
3.1.1: Verify token and privilege
3.1.2: Create Unit bill (values)
3.1.3: Update succeed (unit_bill_id)
4: Unit Bill created

Figure 3 Bill Unit Creation

If a unit bill is not set for a particular month then administer will be notified because without is bill for that month cannot be available. Once the unit bill values are set , it will be display to all the user via "My bill" tab of their dashboard. When administrator creates a new unit bill using the system panel of dashboard (figure 4) the following JSON request message received at keystone along with the token which is generated after successful login of admin

```
{'_apiresource': <Biller
                    {
                         u'vcpu': u'0.005',
                          u'ram': u'0.0083',
                         u'vdisk': u'0.0003',
                          u'enabled': True,
                          u'date': u'2012-04-01 00:00:00',
                          u'changed_on': u'2012-04-26 11:16:27'
                    }
>}
```



Create Cost per Unit

virtual CPU (1 - $): 0.005

RAM (1 MB - $): 0.0083

virtual Disk (1 GB - $): 0.0003

DATE: May 1 2012

Description:

From here you can create a new unit price for resources (aka project).

Figure 4 Creating Bill Unit - Form

Keystone checks the validity of the token attached with this message and creates a new entry in the database. After a successful entry keystone sends back the response to horizon with the unit bill id.

Administrator has the flexibility to create unit bill for upcoming month in advance and it can be edited till corresponding month starts. Anyhow admin cannot be able to alter the unit bill for previous months and current month; it will give the stability for billing.

2.2 Creating instances vies bills

As mentioned earlier in this chapter, the values stored represent the total cost and usage of resources by instances throughout its lift time. So the calculations are made periodically as well as need based and update are made in database so that user and administer will always get the exact instances cost at any time. The default implementation of nova, stores the usage of instances for every month individually. At the beginning of every month horizon get the instance usage data for previous month (if any) from nova-api and calculates the value to be stored in keystone. These values are send to keyston with proper request for updating the database. Below sequence diagram (figure 5) show the message passed during this update.

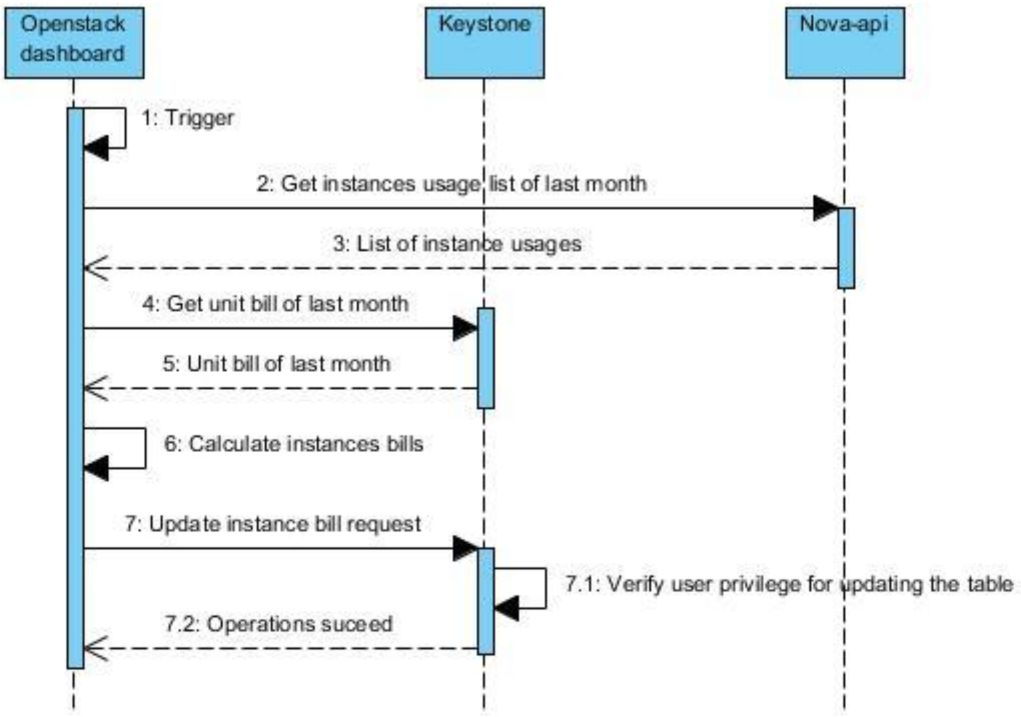

Openstack dashboard
Keystone
Nova-api
1: Trigger
2: Get instances usage list of last month
3: List of instance usages
4: Get unit bill of last month
5: Unit bill of last month
6: Calculate instances bills
7: Update instance bill request
7.1: Verify user privilege for updating the table
7.2: Operations suceed

Figure 5 Instance bill updating

Sample JSON message content of an instance in usage list is

{

u'uptime': 17885, u'disk_size': 99, u'name': u'instance 3', u'user_id': u'user 1', u'tenant_id': u'4', u'hostname': u'instance-3', u'ram_size': 256, u'state': u'active', u'vcpus': 1, u'hours': 4.968189686111111, u'host': u'openstack2', u'image_ref': u'2', u'ended_at': None, u'flavor': u'm1.vsmal', u'started_at': u'2012-04-26 06:04:42', u'id': 3

}

The received message has list of instance for the nova-api and each of it is similar to above format. Horizon will request keystone for the unit bill cost if the previous message is successful. The receive unit bill JSON message will be similar to

{

'_apiresource': <Biller {u'vcpu': u'0.005', u'ram': u'0.0083', u'enabled': True, u'vdisk': u'0.0003', u'date': u'2012-04-01 00:00:00', u'id': u'1', u'changed_on': u'2012-04-26 11:16:27

'}>

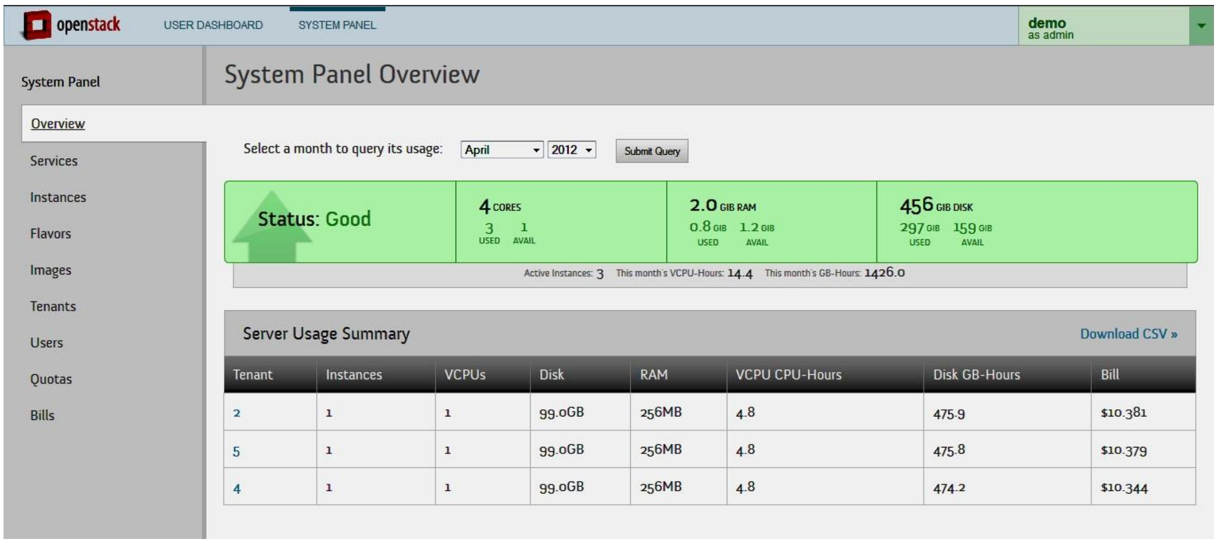
If these two operations succeeded then instance vise bill is calculate and a request is send to the keystone in the following format along with the token values

```
{
        'biller':
        {
                'name': u'instance 2',
                 'total_ram': 11.323350250382223,
                 'total_vcpu': 0.02664568488888889,
                 'total_cost': 11.508271303511112,
                'enabled': True,
                'id': 2,
                 'changed_on': datetime.datetime(2012, 4, 26, 16, 53, 21, 27852),
                 'total_vdisk': 0.15827536824
        }
}
```

After receiving this message along with token, it verifies the token then makes the update if it is valid and corresponding user has enough privilege. A positive reply message is send back to horizon if the operation succeeds. . Figure 6 shows the system panel view showing instance bills of instances for corresponding month.

Apart from above operations to store the lifetime cost of instances, month wise instances is also calculated and display in user panel and system panel. This helps the user and administrator to monitor current month usages of instances.

Instance bill is updated when it is terminated and the enable filed which indicate the status of instance is set to zero (inactive).



System Panel Overview

Select a month to query its usage: April 2012

Status: Good

4 CORES — 3 USED, 1 AVAIL

2.0 GIB RAM — 0.8 GIB USED, 1.2 GIB AVAIL

456 GIB DISK — 297 GIB USED, 159 GIB AVAIL

Active Instances: 3 This month's VCPU-Hours: 14.4 This month's GB-Hours: 1426.0

Server Usage Summary

Tenant	Instances	VCPUs	Disk	RAM	VCPU CPU-Hours	Disk GB-Hours	Bill
2	1	1	99.0GB	256MB	4.8	475.9	$10.381
5	1	1	99.0GB	256MB	4.8	475.8	$10.379
4	1	1	99.0GB	256MB	4.8	474.2	$10.344

Figure 6 Instances and corresponding bill for April 2012

2.3 Creating Tenant Bill

This bill is created when the any users of corresponding tenant launches his/her first instance in OpenStack compute. As like instance bill this is also calculated and updated periodically as well as need based to provide more accurate number to the tenants and administrator Figure 7 show the sequence of messages exchanged between horizon, keystone and nova-api

At the beginning of every month, previous month bill will be created. Horizon triggers this operation and sends request to nova-api to get the list of tenants and its usages for last month. If this operation is succeed, a request is send to keystone for getting the unit cost of resources. Then total bill is calculated for each tenant for updating individually in keystone database.

Along with the token value, the calculated values are send to keystone as request message for updating the database. A sample of this message is shown in the next page.

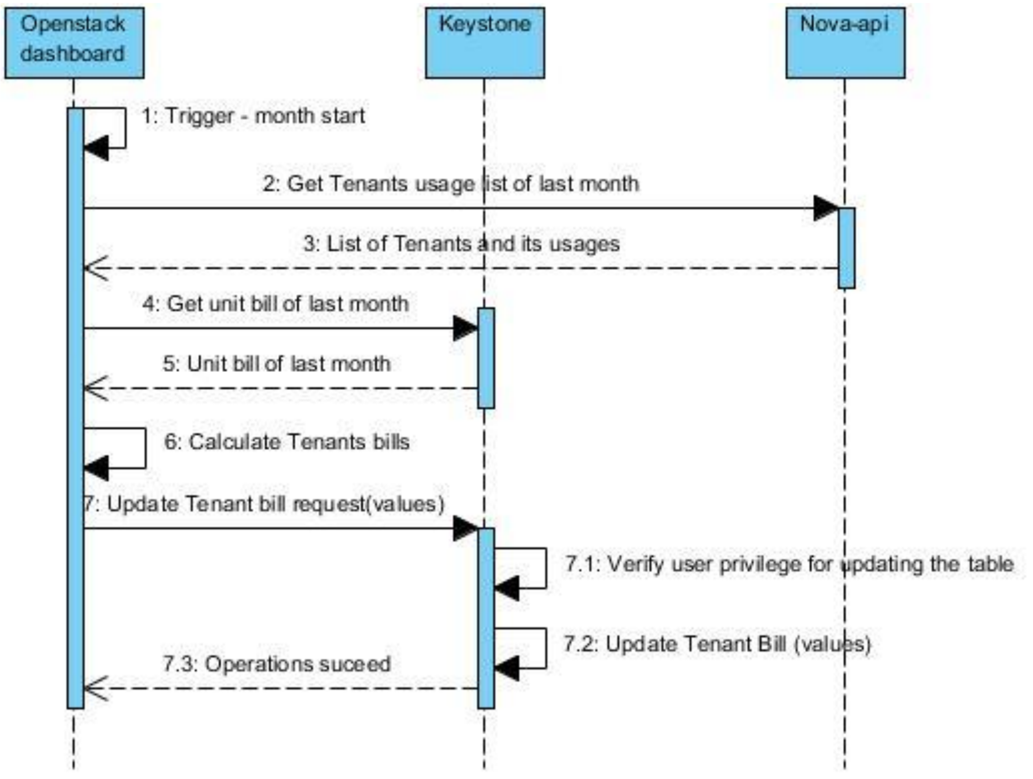
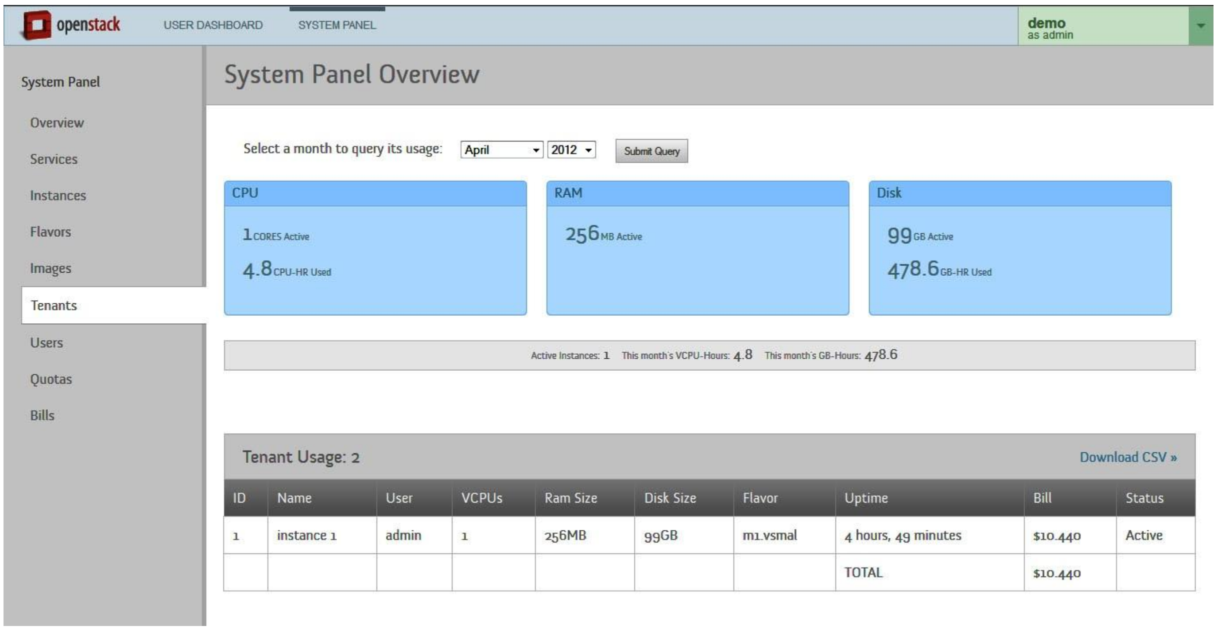

Openstack dashboard
Keystone
Nova-api
1: Trigger - month start
2: Get Tenants usage list of last month
3: List of Tenants and its usages
4: Get unit bill of last month
5: Unit bill of last month
6: Calculate Tenants bills
7: Update Tenant bill request(values)
7.1: Verify user privilege for updating the table
7.2: Update Tenant Bill (values)
7.3: Operations suceed

Figure 7 Tenant bill creation sequence diagram.

```
{       'biller':

        {

                'tenant_id': u'4',

                'total_ram': 9.889859808995555,

                total_vcpu': 0.023272448722222222,

                'total_cost': 10.051370603127777,

                'bill_month': datetime.datetime(2012, 4, 1, 0, 0),

                 'enabled': True,

                 'id': None,

                'total_vdisk': 9.889859808995555

        }

}
```

The token along with this message is verified and privilege of corresponding user is check to see whether he/she has permission to modify the database. If true then the corresponding value is updated in the database and sends back the operation succeed message to horizon. This tenant bill is available for users and system panels. Figure 8 shows the system panel view of tenants and its corresponding instances bill values



System Panel Overview

Select a month to query its usage: April 2012

CPU: 1 CORES Active, 4.8 CPU-HR Used

RAM: 256 MB Active

Disk: 99 GB Active, 478.6 GB-HR Used

Active Instances: 1 This month's VCPU-Hours: 4.8 This month's GB-Hours: 478.6

Tenant Usage: 2

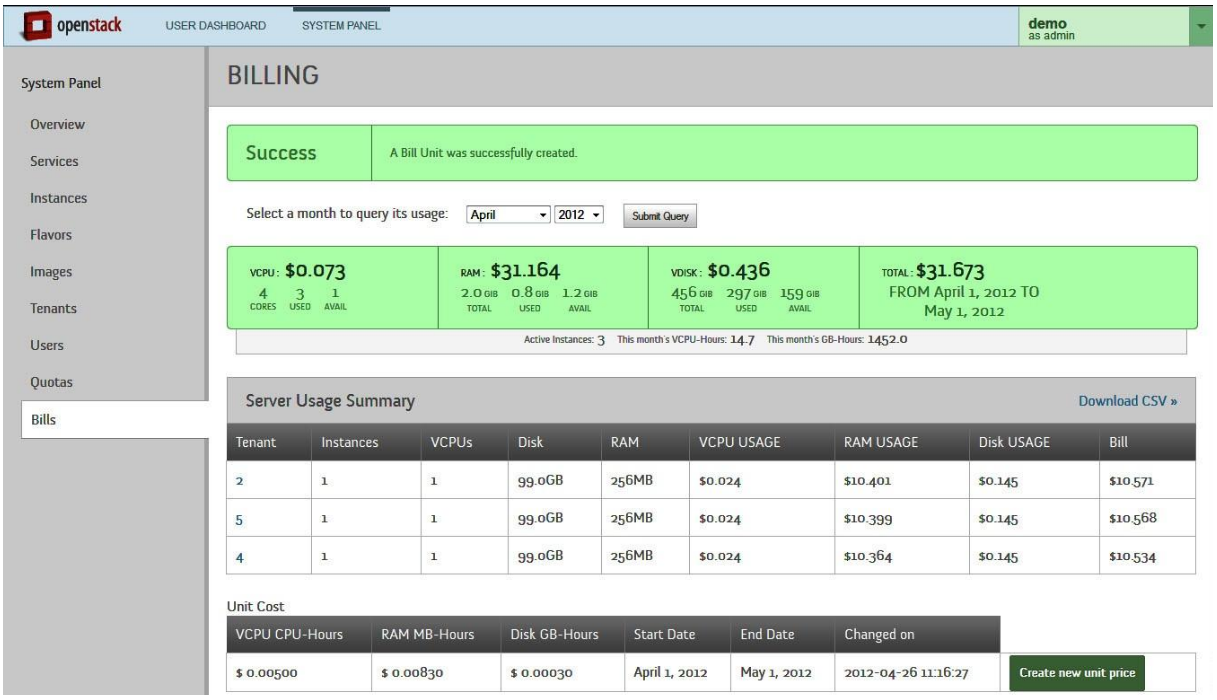
ID	Name	User	VCPUs	Ram Size	Disk Size	Flavor	Uptime	Bill	Status
1	instance 1	admin	1	256MB	99GB	m1.vsmal	4 hours, 49 minutes	$10.440	Active
							TOTAL	$10.440	

Figure 8 System panel -Tenant usage

2.4 Billing Centre

Figure 9 shows the page available for the system administrator to view overall details about billing. Details available in this page are overall usage of the entire OpenStack cloud, instance vise bill, unit cost of resources, link for creating new bill unit for coming months and link for downloadable summary of total cloud usage, tenant usage, unit cost, etc,. Figure 9 show the Billing Centre used by cloud administrators.

‘My Bill’ is made available on the user dashboard so that users can view running cost of the instances lunched by their tenant group. In addition the unit bill for the current month is also made visible on user side for the purpose of mutual verification. Details of previous month bill will also provided on tenant vies as soon as administrator approves the bill. Payment options can also be include in this page if needed.



BILLING

Success A Bill Unit was successfully created.

Select a month to query its usage: April 2012

VCPU : $0.073 — 4 CORES, 3 USED, 1 AVAIL

RAM : $31.164 — 2.0 GIB TOTAL, 0.8 GIB USED, 1.2 GIB AVAIL

VDISK : $0.436 — 456 GIB TOTAL, 297 GIB USED, 159 GIB AVAIL

TOTAL : $31.673 FROM April 1, 2012 TO May 1, 2012

Active Instances: 3 This month's VCPU-Hours: 14.7 This month's GB-Hours: 1452.0

Server Usage Summary

Tenant	Instances	VCPUs	Disk	RAM	VCPU USAGE	RAM USAGE	Disk USAGE	Bill
2	1	1	99.0GB	256MB	$0.024	$10.401	$0.145	$10.571
5	1	1	99.0GB	256MB	$0.024	$10.399	$0.145	$10.568
4	1	1	99.0GB	256MB	$0.024	$10.364	$0.145	$10.534

Unit Cost

VCPU CPU-Hours	RAM MB-Hours	Disk GB-Hours	Start Date	End Date	Changed on
$ 0.00500	$ 0.00830	$ 0.00030	April 1, 2012	May 1, 2012	2012-04-26 11:16:27

Figure 9 Billing centre of system panel

A detailed bill report (.csv) can also be downloaded. A sample of downloaded report is shown below.

```
Usage Report For Period :, apr. 01 2012 00:00,/,apr. 27 2012 10:07
Tenant ID               : ,4
Total Active VCPUs      : ,1
CPU-HRs Used            : ,28.0448738472
Total Active Ram (MB)  : ,256
Total Disk Size         : ,99
Total Disk Usage        : ,2776.44251087

This month's Unit Cost
VCPU/Hr,RAM 1MB/Hr,DISK 1GB/Hr
$0.005,$0.0083,$0.0003

Name,UserId,VCPUs,RamMB,DiskGB,Flavor,Usage(Hours),Uptime(Seconds),State,Cost
instance
instance 3,user 1,1,256,99,m1.vsmal,28.0882366222,16157,Active,$60.6565469857
Total Cost ,,,,,,,,$ 60.6565469857
```

***** This document is a Draft, Final version will be available soon**** **

Summary of entire cloud is available to administrator.

```
Usage Report For Period:,apr. 01 2012 00:00,/,apr. 27 2012 10:09
Active Instances            :,3
This month's VCPU-Hours :,84.3
This month's GB-Hours      :,8345.5

This month's Unit Cost
VCPU/Hr,RAM 1MB/Hr,DISK 1GB/Hr
$0.005,$0.0083,$0.0003

Name,UserId,VCPUs,RamMB,DiskGB,Flavor,Usage(Hours),Uptime(Seconds),State,Cost
instance 1,admin,1,256,99,m1.vsmal,28.1054588444,16219,Active,$60.6937383746
instance 2,user2,1,256,99,m1.vsmal,28.1043477333,16215,Active,$60.6913389301
instance 3,user 1,1,256,99,m1.vsmal,28.0882366222,16157,Active,$60.6565469857



Total Cost ,,,,,,,,,$182.04162429
```

NOTE : *The automatic trigger which is mentioned in this document is not implemented in github.com/ashokcse/openstack-bill

**The calculation of bills are event driven i.e calculated whenever user or admin need, not automatically as mentioned in this document