The relationship between endpoints, regions, etc in keystone OpenStack

Asked 11 years, 2 months ago Modified 7 years, 6 months ago

Viewed 14k times

20

I'm really trying to understand the under the hood of keystone regarding the relationships among endpoints, regions, tenants, services, users and roles. I've tried to find the related documents but sadly, failed.

Could anybody give any pointers or explanations?

openstack keystone

Share

Improve this question

Follow

asked Sep 25, 2013 at 12:08

jaeyong

9,323 ● 15 ● 52 ● 64

4 Answers

Sorted by: Highest score (default)

36

Keystone is the identity management service for OpenStack.

Essentially it's role is to grant tokens to users be they people, services, or anything at all.

If you make an API query anywhere in OpenStack, keystone's API is how it is discovered if you are allowed to make that API query.

Let's work our way up from the ground.

Users. Users in Keystone today are generally people. There isn't enough fine grained ACL support at this moment to really call many of the users in OpenStack a 'service' account in a traditional sense. But there is a service account that is used as a backhaul connection to the Keystone API as part of the OpenStack infrastructure itself. We'll avoid delving into that anomalous user.

When a user authenticates to Keystone (you hit up the OS_AUTH_URL to talk to keystone.. usually port 5000 of the keystone api box), the user says hey " I am user X , I have password Y, and I belong to tenant Z".

X can be a username or userid (unique uuid of user) Y is a password, but you can authenticate with a token as well. Z is a tenant name or tenant id (unique uuid of tenant). in past Keystone APIs you didn't NEED to specify a tenant name, but your token wouldn't be very useful if you didn't as the token wouldn't be associated with your tenant and you would then be denied any ACLs on that tenant.

So... a user is a fairly obvious thing. A password is a fairly obvious thing. But what's a tenant?

Well a tenant is also known as a project. In fact, there have been repeated attempts to make the name be either tenant or project, but as a result of an inability to stick to just one term they both mean the same thing. As far as the API is concerned a project IS a tenant. So if you log into horizon you will see a drop down for your projects. Each project corresponds to a tenant id. Your tokens are associated with a specific tenant id as well. So you may need several tokens for a user if you intend to work on several tenants the user is attached to.

Now, say you add a user to the tenant id of admin. Does that user get admin privileges? The answer is no. That's where roles come into play. While the user in the admin tenant may have access to admin virtual machines and quotas for spinning up virtual machines that user wouldn't be able to do things like query keystone for a user list. But if you add an admin role to that user, they will be endowed with the ACL rights to act as an admin in the keystone API, and other APIs. So think of a tenant as a sort of resource group, and roles as an ACL set.

regions are more like ways to geographically group physical resources in the openstack infrastructure environment. say you have two segmented data centers. you might put one in region A of your openstack environment and another in region B. regions in terms of their usefulness are quickly evolving, especially with the introduction of cells and domains in more recent openstack releases. You probably don't need to be a

master of this knowledge unless you intend to be architecting large clouds.

keystone provides one last useful thing. the catalog. the keystone catalog is kind of like the phone book for the openstack APIs. whenever you use a command line client, like when you might call nova list to list your instances, nova first authenticates to keystone and gets you a token to use the API, but it also immediately asks keystone catalog for a list of API endpoints. For keystone, cinder, nova, glance, swift... etc. nova will really only use the nova-api endpoint, though depending on your query you may use the keystone administrative API endpoint.... we'll get back to that. But essentially the catalog is a canonical source of information for where APIs are in the world. That way you only ever need to tell a client where the public API endpoint of keystone is, and it can figure out the rest from the catalog.

Now, I've made reference to the public API, and the administrative API for keystone.

Yep keystone has two APIs... sort of. It runs an API on port 5000 and another one up in the 32000 range. The 5000 is the public port. This is where you do things like find the catalog, and ask for a token so you can talk to other APIs. It's very simple, and somewhat hardened. The administrative API would be used for things like changing a users password, or adding a new role to a user.

Pretty straight forward?

Share Improve this answer

Follow

answered Sep 25, 2013 at 16:59

Matt Joyce

2,010 ● 2 ● 20 ● 32

Thanks! Couple of follow up questions: You said that cinder, nova, glance stuffs use nova-api endpoint. So, does it mean that endpoint is like an API? And, if I list up endpoint-list it has service_id field. Is this the one that points to something like nova-api? Also, endpoint-list shows publicurl, internalurl, and adminurl. Could you briefly explain what are they?
– jaeyong Sep 26, 2013 at 0:30

More questions: How can I associate a role and a service endpoints? I tried to find the relations using keystone xxx-list and no foriegn keys between them. – jaeyong Sep 26, 2013 at 2:25

4

A late reply but I hope it helps future readers.

endpoints are like the point of contact for you to use a service. adminurl as the name says is for admin users only. internal url is for the various services to talk to each other if they have to and public url is for anyone else to use.

Why do you need separate public and internal url is explained here.

You do not have to assign a role for service endpoints. You assign roles to the users who access the service and this is how you do it.

Share Improve this answer

Follow

edited May 8, 2016 at 5:52

answered Jan 12, 2014 at 13:20

Akilesh

1,300 ● 1 ● 9 ● 17

1 A lot changed in Openstack documentation since then. I have updated the relevant links from the liberty Openstack installation guide. – Akilesh May 8, 2016 at 5:52

3

This is my way of understanding tenant, user, role, permission in openstack keystone. you may find it interesting too.

Share Improve this answer

Follow

answered Feb 12, 2014 at 1:47

Prosunjit Biswas

1,101 ● 11 ● 17

3

I will try to put this in layman's language.

Service - Openstack needs large number of services to get the cloud infrastructure (Compute,Storage and Network) going. To enable them smoothly at the same time to have fine grained control, Openstack uses the notion of services. Services enable end users to manipulate one of these 3 core resources. For example: For storage, cinder and swift services are used. These

services further can be configured to use Ceph or gluster in backend.

Endpoint- The point where you 'get in' to an openstack service to do something useful or destructive. A service may be running, but to 'get in', an endpoint is needed, which will look like http://my-fancy-IP:hard-to-remember-port-number/v3.0. So, no endpoints created in Openstack system for a particular openstack service? Not possible to access that service.

Region - Nothing to do with geography or location. A term used for dividing/grouping full openstack deployments, if you happen to have many of them. Division may be based on the location of physical servers.

User - End user or You.

Project/Tenant - A container to separate your resources (compute,storage network)

Domain - Group of projects, groups and users. Provides separation of resources for users and can accommodate multiple projects. A user with domain admin role can create/delete/update any project, users, groups etc in that particular domain.

Role - Your rights to perform any operation on openstack services. Think of role, Think of user.

Token - It is an ID (a long string) given by keystone to you. You can access any openstack service according to

the role assigned to you using this token.

For example, you can request nova and say, hey nova I got token1 from keystone. I want to delete 'don't-ever-delete-me-ever' server. Nova takes token1 and says, hey keystone, this user has token1 and wants to delete the server 'don't-ever-delete-me-ever'.Keystone, looks at token1 and according to the role assigned to user, will say, ok nova allow/don't allow user to delete 'don't-ever-delete-me-ever' server.

Share Improve this answer

Follow

answered Jun 22, 2017 at 8:16

pumpkin_cat

1,256 ● 14 ● 17