chourouk:

éna je te dis cette mission, la clé pour la réussir, c’est de faire des connaissances dans les équipes avec lesquelles tu travailles

Bouygues c’est 80% process

le plus important c’est de vite connaitre les process, essayer de connaitre des personnes d’autres équipes, les contacter sur teams ou skype pour accélérer les choses des fois

et tu auras Oussama Soualah

c’est vraiment quelqu’un de tres bien

mais alhamdolillAh, tkt surtout pas tu vas gérer

=====================================

# openstack service list: c'est les services installé

#openstack endpoint list c'est les "interfaces" généralement 3: Admin internal w external

se sont des apis.

======================================

# sudo podman ps - -filter status=running - - format=” {{.ID}} {{.Names}} {{.Status}} ” | grep

keystone

# sudo podman ps - -filter status=running - - format=” table {{.ID}} {{.Names}} {{.Status}} ” | grep

keystone

# sudo podman stats cinder\_api ⇒ donne les statistique de consommation de ce conteneur en temps réel

# sudo podman logs cinder\_api ⇒ peut donner les logs de ce conteneur

# sudo podman images ⇒ donnes les caches des images en locale

# sudo podman logs –since 3h –tail 4 container-name

# sudo podman exec -it container-name /bin/bash

#(dans le conteneur on exécute) ps aux ⇒ donne les différents processus qui tournent dans le conteneur

#sudo podman exec -it cinder\_api hostname

La commande podman stats affiche des statistiques d’utilisation des ressources de conteneur en temps réel. La commande stats est similaire à l’utilisation de la commande top Linux. Spécifiez une liste de conteneurs séparés par des espaces pour interroger plusieurs conteneurs. Les conteneurs indiquant Stopped ou Exited ne renverront pas de données.

La commande podman inspect affiche la configuration du conteneur au format JSON. Utilisez la commande jq pour filtrer ou extraire des données de la sortie JSON.

Pour vérifier le statut d’un service en conteneur, exécutez la commande systemctl status :

**# sudo systemctl status tripleo\_cinder\_api**

Pour afficher la manière dont systemd gère le conteneur, affichez le fichier d’unité du service : # **cat /etc/systemd/system/tripleo\_cinder\_api.service**

Pour arrêter (stop), démarrer (start) ou redémarrer (restart) un service en conteneur, utilisez les commandes systemctl :

#sudo systemctl status tripleo\_cinder\_api

#sudo systemctl start tripleo\_cinder\_api

#sudo systemctl stop tripleo\_cinder\_api

#sudo systemctl restart tripleo\_cinder\_api

**NB:** Systemd monitors container health checks, which are not displayed by podman ps, using systemd timers. To list containers timers, use the systemctl list-timers command. This example filters to view only tripleo unit services:

**[heat-admin@controller ~]$ sudo systemctl list-timers | grep tripleo**

To verify a specific container timer, use the systemctl status command:

**[heat-admin@controller ~]$ sudo systemctl status tripleo\_cinder\_api\_healthcheck.timer**

### **Log and Configuration File Locations**

In the latest version of RHOSP, the standard output (stdout) and standard errors (stderr) are consolidated in a single file per container, located in the /var/log/containers/stdouts directory. Life-cycle tools manage containers to gather these outputs into log collections, including deleted containers, in the /var/log/containers/stdouts directory.

**[heat-admin@controller ~]$ sudo ls -1 /var/log/containers/stdouts**

#### **Configuration File Location**

In RHOSP versions with containerized services, container configuration files are located in the /var/lib/config-data/puppet-generated/container\_name directory.

[heat-admin@controller ~]$ sudo ls -1 /var/lib/config-data/puppet-generated/

The systemd unit files of containerized services are named with a tripleo\_ prefix, because they are installed by TripleO.

[heat-admin@controller ~]$ **sudo ls /etc/systemd/system/tripleo\_\*.service**

**================================================**

dans undercloud :

# openstack endpoint list -c ‘Service type’ -c ‘Interface’ -c ‘URL’

#openstack service list