Build Openshift PaaS using RHEV virtualization (and run 20k pods)

Elvir Kurić

Red Hat Performance Engineering

Devconf 2017, BRNO, Czechia

Agenda

- Motivation
- Technical specification
- Installing OpenShift on top of RHEV virtualization
- Scheduling 20k pods across 210 Openshift nodes
- Critical setup places
- Q/A

Motivation

- Why OpenShift Container Platform on top of RHEV?
  - We can make that (and it is fully supported)
  - use case - OCP on RHEV

Motivation

- It was project task within Performance team
- Target was to schedule as many as possible pods

RHEV Hardware Specification

- 19 RHEV hypervisors
  - 14 machines 132 GB of RAM , 32 CPUs
  - 5 machines 132 GB of RAM , 48 CPUs
- In total 688 cores and 2.5 TB of RAM
- 10Gb network for OCP traffic (RHEV logical net)
- iSCSI storage domain
- EMC storage in backend (VNXe1600)

RHEV Installation

- kickstart RHEL
- On top of RHEL installed RHEV packages
- Other install option is to use directly RHEV ISOs

RHEV Virtual Machines Creation

- Python scripts to create VMs

Use API - Python ovirtsdk

- At scale of hundreds of virtual machines not easy way to create them without using API
- Ovirtsdk offers nice features

http://www.ovirt.org/develop/release-management/features/infra/python-sdk/

https://github.com/ekuric/openshift/blob/master/_rhev/vm_create.md

RHEV Virtual Machines Creation

- RHEV will use by default **thin / sparse** allocation for storage for VMs
- Second option is Preallocated storage

RHEV Virtual Machines Creation

- Use preallocated storage type at least for masters/etcd/routers
- In this installation nodes were using thin lvm storage type

RHEV Virtual Machines Creation

- With ovirtsdk API
  disks=params.Disks(clone=True)
  - This will cost additional storage space, but will help with performances
  - https://access.redhat.com/documentation/en/red-hat-virtualization/4.0/single/administration-guide/#Understanding_virtual_disks

RHEV Virtual Machines Creation

- Use preallocated storage type for masters/etcd/routers
- In this installation nodes were using thin lvm storage for VMs
- Masters/ETCDs/Infra nodes were build on top of Preallocated disks

Install OCP

- In this specific configuration there were
  - 3 Master servers
  - 3 ETCD servers
  - 1 lb
  - 7 Infra Nodes (for router/registry/metrics/logging pods)
  - 210 nodes for applications
  - In total 224 OCP machines

Install OCP

- Etcd servers not collocated with openshift masters and with only etcd service running (BZ 1387149 - iptables)
- Lb self standing machine

Install OCP

- Masters - not schedulable
- Infra nodes only for hosting infra pods (router / registry / metrics / logging)
- General purpose nodes - for all other pods

Install OCP in 3 steps

- Create VMs using API [1]
  https://github.com/ekuric/openshift/blob/master/_rhev/vm_create.md
- Collect ips/fqdn of VMs [2]

https://github.com/ekuric/openshift/blob/master/_rhev/vm_create.md

- Use openshift ansible [3]
  https://github.com/openshift/openshift-ansible

Install OCP

- Feed ips/fqdn into inventory file
- Ansible-playbook -i inventory openshift-ansible/config.yml
- Get coffee (or two) - depending on software preloaded on VMs installation can take some time for 200+ OCP machines

Speed up Install time of OCP

- **Increase ansible forks** /etc/ansible/ansible.cfg
- Prepare template with as much as possible stuff in advance to avoid download during install
- Install nfs / gluster/ ceph packages, setup up Selinux booleans in advance on gold image

Create PODs

- If all went fine, cluster should be up
- # oc get nodes | wc -l

oc get nodes | wc -l

224

Create Pods

- Cluster-loader - to hit 20k pods

Link : https://github.com/openshift/svt

- pods-per-core": ["0"] - to allow more than 10 pods per core (to override default)

Create Pods

- No special requirements on pod type was imposed - I just needed many pods

Lessons learned

- Storage used for RHEV storage domain has to be high end - at least for numbers we talk here about
- etcd/masters/infra machines - put them on “preallocated” RHEV disk
- Etcd is sensitive on delays - plan accordingly to have VMs with fast disk / enough RAM

Lessons learned - ETCD

- On Etcd nodes install only ‘etcd’ and no docker/master/node (BZ 1387149)
- Ensure time on OCP/RHEV is in sync
- Watch etcd/master logs

Lessons learned - ETCD

- Watch etcd/master logs

---- --- ---- ---

Sep 27 00:04:01 dhcp7-237 etcd: failed to send out heartbeat on time (deadline exceeded for 1.766957688s)

Sep 27 00:04:01 dhcp7-237 etcd: server is likely overloaded

Lessons learned - ETCD

Playing with ETCD timers , could help…

ETCD_ELECTION_TIMEOUT=

ETCD_HEARTBEAT_INTERVAL=

Lessons learned - ETCD

- Check https://coreos.com/etcd/docs/latest/tuning.html

Acceptance test!

- Scenario: restart cluster and relocate HW
- When starting VMs(=nodes) give at least 15s between VMs start - in case of high end storage used for storage domain, this might not be the case

Motivation bonus

- Example of Red Hat's Technologies (RHEL + RHEV + OCP + Ansible) building PaaS
- Other way could be : RHEL + OSP + OCP + Ansible

Links

- Presentation https://github.com/ekuric/_talks/tree/master/devconf

Q/A

Thank you

ekuric@redhat.com