Kubernetes Bare-metal Testing

This document describes the initial testing efforts for the Kubernetes bare-metal testbed in the OPNFV Lab.

Introduction

With the ongoing efforts to define and deploy a Kubernetes bare-metal testbed in the OPNFV Lab, there is a need for some method to test and verify that the Kubernetes cluster plus additional features are installed and working correctly. Instead of running a full test suite intended for verification and compliance of commercial products, the intention here is to start out with basic checks and tests cases that can be expanded in parallel with the work that is being done for the bare-metal testbed.

Tools

As a starting point, testing is done to verify the network connectivity through the SR-IOV Network Device Plugin, which is deployed and configured as part of the BMRA Kubernetes installer used for the bare-metal testbed.

The tools chosen for this verification is **PROX** (Packet pROcessing eXecution engine) and **rapid** (Rapid Automated Performance Indication for Dataplane), both available as part of the OPNFV SampleVNF project.

- PROX is the network function that is running inside of a container. The functionality depends on the configuration, which supports packet/traffic generation, forwarding and impair (drop, delay) functionality
- rapid is the framework for running benchmarks and tests. A high level configuration is provided to rapid, and depending on the test cases each instance of PROX will be configured accordingly. Rapid also manages the pass/fail criteria, usually depending on metrics such as throughput, droprate, and min/max/percentile latency measurements.

Installation

The installation of PROX and rapid assumes that a Kubernetes cluster has been provisioned using BMRA. The steps for provisioning a cluster can be found here.

In order to avoid having to do additional network configuration for the cluster, the installation of PROX and rapid should happen on one of the master nodes.

Start by cloning the code and building the PROX image:

```
$ git clone https://github.com/opnfv/samplevnf.git
$ cd samplevnf/VNFs/DPPD-PROX/helper-scripts/rapid
$ ./dockerimage.sh build
```

At this point there are two options. (1) Configure a local registry on the master node, and have worker nodes pull the image when needed, or (2) copy the newly created prox_slim.tar file to worker nodes, and load the image on each of the workers using `docker load -i prox_slim.tar`. For this example option (1) will be used, as it simplifies the process of updating the image if changes are needed, at the cost of a few extra steps needed to configure the registry.

On the master node:

```
$ docker run -d -p 5000:5000 --name registry registry:2
  ## Taken from https://docs.docker.com/registry/

$ ./dockerimage.sh push
```

On each of the worker nodes:

```
$ vim /etc/systemd/system/docker.service.d/docker-options.conf
  ## Add `--insecure-registry=<IP of master node>:5000 \` to the list of input parameters (in "Environment")

$ systemctl daemon-reload
$ systemctl restart docker
```

While the next step is optional, it is recommended to update the CPU governor on the worker nodes to use a different profile, as it defaults to "balanced. Due to the nature of the tools and testing, this should be changed to "network-latency".

On the worker nodes:

```
$ tuned-adm profile network-latency
```

At this point the PROX image will be available on all the workers. The next steps will prepare for, and run, PROX instances based on the configuration.

On the master node:

```
$ pip install paramiko
$ yum install python3
$ pip3 install future
$ vim rapid.pods
  ## Update the nodeSelector to match the cluster and location of PROX containers
  ## You can get these using `kubectl get nodes`

$ vim pod-rapid.yaml
  ## Remove the annotations:
    -  annotations:
    - k8s.v1.cni.cncf.io/networks: intel-sriov-vfio
  ## Update the image location:
    - image: localhost:5000/prox_slim:latest
    + image: <IP of master node>:5000/prox_slim:latest
  ## Update the number of hugepages:
    - hugepages-2Mi: 512Mi
    + hugepages-2Mi: 1Gi
  ## Update the VF resource requests/limits
    - intel.com/intel_sriov_vfio: '1'
    + intel.com/intel_sriov_dpdk: '1'
    ## If you modified `sriovdp_config_data` during BMRA installation, the name might be different.
      $ yum install jq
      $ kubectl get node <node> -o json | jq ".status.allocatable"

$ ./createrapidk8s.py
```

At this point there should be a set of PROX containers running. This can be verified through `kubectl get pods`, which should show a set of pods named "pod-rapid-#".

Running tests

Rapid will default to one of the provided test configurations, `basicrapid.test`, which is the one that will be used to verify connectivity between nodes and containers in the cluster. For now the default configuration can be left unchanged, but if performance benchmarks are relevant then the test configuration should be updated to include additional resources and tweaks related to the test cases.

Running the default test with rapid from the master node:

```
$ ./runrapid.py
```