



Red Hat OpenStack Platform 12

Network Functions Virtualization Planning and Configuration Guide

Planning and Configuring the Network Functions Virtualization (NFV) OpenStack Deployment

Red Hat OpenStack Platform 12 Network Functions Virtualization Planning and Configuration Guide

Planning and Configuring the Network Functions Virtualization (NFV) OpenStack Deployment

OpenStack Team
rhos-docs@redhat.com

Legal Notice

Copyright © 2018 Red Hat, Inc.

The text of and illustrations in this document are licensed by Red Hat under a Creative Commons Attribution-Share Alike 3.0 Unported license ("CC-BY-SA"). An explanation of CC-BY-SA is available at

<http://creativecommons.org/licenses/by-sa/3.0/>

. In accordance with CC-BY-SA, if you distribute this document or an adaptation of it, you must provide the URL for the original version.

Red Hat, as the licensor of this document, waives the right to enforce, and agrees not to assert, Section 4d of CC-BY-SA to the fullest extent permitted by applicable law.

Red Hat, Red Hat Enterprise Linux, the Shadowman logo, JBoss, OpenShift, Fedora, the Infinity logo, and RHCE are trademarks of Red Hat, Inc., registered in the United States and other countries.

Linux ® is the registered trademark of Linus Torvalds in the United States and other countries.

Java ® is a registered trademark of Oracle and/or its affiliates.

XFS ® is a trademark of Silicon Graphics International Corp. or its subsidiaries in the United States and/or other countries.

MySQL ® is a registered trademark of MySQL AB in the United States, the European Union and other countries.

Node.js ® is an official trademark of Joyent. Red Hat Software Collections is not formally related to or endorsed by the official Joyent Node.js open source or commercial project.

The OpenStack ® Word Mark and OpenStack logo are either registered trademarks/service marks or trademarks/service marks of the OpenStack Foundation, in the United States and other countries and are used with the OpenStack Foundation's permission. We are not affiliated with, endorsed or sponsored by the OpenStack Foundation, or the OpenStack community.

All other trademarks are the property of their respective owners.

Abstract

This guide contains important planning information and describes the configuration procedures for SR-IOV and OVS-DPDK in your Red Hat OpenStack Platform with NFV deployment.

Table of Contents

PREFACE	5
CHAPTER 1. OVERVIEW	6
CHAPTER 2. HARDWARE REQUIREMENTS	7
2.1. TESTED NICS	7
2.2. DISCOVERING YOUR NUMA NODE TOPOLOGY	7
2.3. REVIEW BIOS SETTINGS	11
CHAPTER 3. SOFTWARE REQUIREMENTS	13
3.1. SUPPORTED CONFIGURATIONS FOR NFV DEPLOYMENTS	13
3.2. SUPPORTED DRIVERS	13
3.3. COMPATIBILITY WITH THIRD PARTY SOFTWARE	13
CHAPTER 4. NETWORK CONSIDERATIONS	14
CHAPTER 5. PLANNING AN SR-IOV DEPLOYMENT	15
5.1. HARDWARE PARTITIONING FOR AN SR-IOV DEPLOYMENT	15
5.2. TOPOLOGY OF AN NFV SR-IOV DEPLOYMENT	15
5.2.1. NFV SR-IOV without HCI	16
CHAPTER 6. CONFIGURING AN SR-IOV DEPLOYMENT	18
6.1. OVERVIEW OF SR-IOV CONFIGURABLE PARAMETERS	18
6.2. CONFIGURING SR-IOV WITH CONTROL PLANE BONDING	19
6.2.1. Creating the ComputeSriov composable role	20
6.2.2. Configuring tuned for CPU affinity	21
6.2.3. Configuring SR-IOV parameters	21
6.2.4. Configuring the Controller node	23
6.2.5. Configuring the Compute node for SR-IOV interfaces	25
6.2.6. Deploying the overcloud	26
6.3. CREATING A FLAVOR AND DEPLOYING AN INSTANCE FOR SR-IOV	27
CHAPTER 7. PLANNING YOUR OVS-DPDK DEPLOYMENT	29
7.1. OVS-DPDK WITH CPU PARTITIONING AND NUMA TOPOLOGY	29
7.2. OVERVIEW OF WORKFLOWS AND DERIVED PARAMETERS	29
7.3. DERIVED OVS-DPDK PARAMETERS	30
7.4. OVERVIEW OF MANUALLY CALCULATED OVS-DPDK PARAMETERS	31
7.4.1. CPU parameters	31
7.4.2. Memory parameters	32
7.4.3. Networking parameters	34
7.4.4. Other parameters	35
7.5. TWO NUMA NODE EXAMPLE OVS-DPDK DEPLOYMENT	35
7.6. TOPOLOGY OF AN NFV OVS-DPDK DEPLOYMENT	36
CHAPTER 8. CONFIGURING AN OVS-DPDK DEPLOYMENT	39
8.1. DERIVING DPDK PARAMETERS WITH WORKFLOWS	39
8.2. OVS-DPDK TOPOLOGY	42
8.3. CONFIGURING TWO-PORT OVS-DPDK DATA PLANE BONDING WITH VLAN TUNNELLING	44
8.3.1. Generating the ComputeOvsDpdk composable role	44
8.3.2. Configuring tuned for CPU affinity	44
8.3.3. Configuring the OVS-DPDK parameters	45
8.3.4. Configuring the Controller node	47
8.3.5. Configuring the Compute node for DPDK interfaces	48
8.3.6. Deploying the overcloud	50

8.4. CONFIGURING OVS-DPDK WITH VXLAN TUNNELLING	50
8.4.1. Generating the ComputeOvsDpdk composable role	50
8.4.2. Configuring tuned for CPU affinity	51
8.4.3. Configuring OVS-DPDK parameters	51
8.4.4. Configuring the Controller node	53
8.4.5. Configuring the Compute node for DPDK interfaces	55
8.4.6. Deploying the overcloud	56
8.5. SETTING THE MTU VALUE FOR OVS-DPDK INTERFACES	56
8.6. CONFIGURING SECURITY GROUPS	58
8.7. SETTING MULTIQUEUE FOR OVS-DPDK INTERFACES	58
8.8. KNOWN LIMITATIONS	59
8.9. CREATING A FLAVOR AND DEPLOYING AN INSTANCE FOR OVS-DPDK	59
8.10. TROUBLESHOOTING THE CONFIGURATION	61
CHAPTER 9. CONFIGURING A HETEROGENEOUS COMPUTE CLUSTER	63
9.1. NAMING CONVENTIONS	63
9.2. CREATING THE SR-IOV AND OVS-DPDK CUSTOM ROLES	64
9.3. CONFIGURING TUNED FOR CPU AFFINITY	66
9.4. DEFINING THE SR-IOV AND OVS-DPDK ROLE-SPECIFIC PARAMETERS	67
9.5. CONFIGURING SR-IOV AND DPDK COMPUTE NODES	69
9.6. DEPLOYING THE OVERCLOUD	72
CHAPTER 10. CONFIGURING SR-IOV AND DPDK INTERFACES ON THE SAME COMPUTE NODE	74
10.1. CREATING THE COMPUTEOVSDPDKSRIOV COMPOSABLE ROLE	74
10.2. CONFIGURING TUNED FOR CPU AFFINITY	75
10.3. DEFINING THE SR-IOV AND OVS-DPDK ROLE-SPECIFIC PARAMETERS	76
10.4. CONFIGURING THE COMPUTE NODE FOR SR-IOV AND DPDK INTERFACES	78
10.5. DEPLOYING THE OVERCLOUD	79
10.6. CREATING A FLAVOR AND DEPLOYING AN INSTANCE WITH SR-IOV AND DPDK INTERFACES	80
CHAPTER 11. UPGRADING RED HAT OPENSTACK PLATFORM WITH NFV	82
CHAPTER 12. PERFORMANCE	83
CHAPTER 13. FINDING MORE INFORMATION	84
APPENDIX A. SAMPLE SR-IOV YAML FILES	85
A.1. SAMPLE SR-IOV TWO-PORT CONTROL PLANE BONDING YAML FILES	85
A.1.1. roles_data.yaml	85
A.1.2. post-install.yaml	89
A.1.3. network.environment.yaml	90
A.1.4. controller.yaml	92
A.1.5. compute-sriov.yaml	95
A.1.6. overcloud_deploy.sh	98
APPENDIX B. SAMPLE OVS-DPDK YAML FILES	100
B.1. SAMPLE VLAN OVS-DPDK DATA PLANE BONDING YAML FILES	100
B.1.1. roles_data.yaml	100
B.1.2. post-install.yaml	103
B.1.3. network.environment.yaml	104
B.1.4. controller.yaml	106
B.1.5. computeovsdpdk.yaml	109
B.1.6. overcloud_deploy.sh	112
B.2. SAMPLE VXLAN OVS-DPDK DATA PLANE BONDING YAML FILES	113
B.2.1. roles_data.yaml	113

B.2.2. post-install.yaml	116
B.2.3. network.environment.yaml	117
B.2.4. controller.yaml	119
B.2.5. compute-ovs-dpdk.yaml	122
B.2.6. overcloud_deploy.sh	125
APPENDIX C. SAMPLE HETEROGENEOUS CLUSTER YAML FILES	127
C.1. SAMPLE SRIOV AND DPDK IN A HETEROGENEOUS COMPUTE CLUSTER YAML FILES	127
C.1.1. roles_data.yaml	127
C.1.2. post-install.yaml	131
C.1.3. network.environment.yaml	132
C.1.4. controller.yaml	135
C.1.5. compute-ovs-dpdk.yaml	138
C.1.6. compute-sriov.yaml	141
C.1.7. overcloud_deploy.sh	144
APPENDIX D. DIFFERENT INTERFACES ON SAME COMPUTE NODE YAML FILES	145
D.1. SAMPLE SR-IOV AND DPDK ON THE SAME COMPUTE NODE YAML FILES	145
D.1.1. roles_data.yaml	145
D.1.2. post-install.yaml	149
D.1.3. network.environment.yaml	149
D.1.4. controller.yaml	151
D.1.5. compute-ovsdpdksriv.yaml	155
D.1.6. overcloud_deploy.sh	158
APPENDIX E. REVISION HISTORY	159

PREFACE

Red Hat OpenStack Platform provides the foundation to build a private or public cloud on top of Red Hat Enterprise Linux. It offers a massively scalable, fault-tolerant platform for the development of cloud enabled workloads.

This guide describes the steps to plan and configure SR-IOV and Open vSwitch with DPDK data path (OVS-DPDK) using the Red Hat OpenStack Platform director for NFV deployments.

CHAPTER 1. OVERVIEW

Network Functions Virtualization (NFV) is a software solution that virtualizes a network function on general purpose, cloud based infrastructure. NFV allows the Communication Service Provider to move away from traditional hardware.

For a high level overview of the NFV concepts, see the [Network Functions Virtualization Product Guide](#).



NOTE

OVS-DPDK and SR-IOV configuration depends on your hardware and topology. This guide provides examples for CPU assignments, memory allocation, and NIC configurations that may vary from your topology and use case.

Red Hat OpenStack Platform director allows you to isolate the overcloud networks. Using this feature, you can separate specific network types (for example, external, tenant, internal API and so on) into isolated networks. You can deploy a network on a single network interface or distributed over a multiple host network interface. Open vSwitch allows you to create bonds by assigning multiple interfaces to a single bridge. Network isolation in a Red Hat OpenStack Platform installation is configured using template files. If you do not provide template files, all the service networks are deployed on the provisioning network. There are three types of template configuration files:

- **network-environment.yaml** - this file contains the network details such as, subnets, IP address ranges that are used for the network configuration on the overcloud nodes. In addition, this file also contains the different settings that override the default parameter values for various scenarios.
- Host network templates (for example, **compute.yaml**, **controller.yaml**) - Define the network interface configuration for the overcloud nodes. The values of the network details are provided by the **network-environment.yaml** file.
- Post install configuration file (**post-install.yaml**) - Provides various post configuration steps, for example:
 - Tuned installation and configuration. The **tuned** package contains the **tuned** daemon that monitors the use of system components and dynamically tunes system settings based on that monitoring information. To provide proper CPU affinity configuration in OVS-DPDK and SR-IOV deployments, you should use the **tuned cpu-partitioning** profile. See the [Performance Tuning Guide](#) for details on this package.

These heat template files are located at `/usr/share/openstack-tripleo-heat-templates/` on the undercloud node. For samples of these heat template files for NFV, see the [Sample YAML Files](#).

The following sections provide more details on how to plan and configure the heat template files for NFV using the Red Hat OpenStack Platform director.



NOTE

NFV configuration makes use of YAML files. See [YAML in a Nutshell](#) for an introduction to the YAML file format.

CHAPTER 2. HARDWARE REQUIREMENTS

This section describes the hardware details necessary for NFV.

You can use [Red Hat Technologies Ecosystem](#) to check for a list of certified hardware, software, cloud provider, component by choosing the category and then selecting the product version.

For a complete list of the certified hardware for Red Hat OpenStack Platform, see [Red Hat OpenStack Platform certified hardware](#).

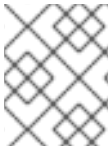
2.1. TESTED NICS

The following hardware have been tested to work with the Red Hat OpenStack Platform 12 NFV deployment:

SR-IOV

Red Hat tested the SR-IOV 10G for Mellanox and Qlogic. Red Hat also tested the following Intel cards:

- 82598, 82599, X520, X540, X550, X710, XL710, X722, XXV710.



NOTE

Red Hat has verified original Intel NICs only and not any other NICs that use the same drivers.

OVS-DPDK

Red Hat tested the following NICs for OVS-DPDK:

Intel

82598, 82599, X520, X540, X550, X710, XL710, X722, XXV710.



NOTE

Red Hat has verified original Intel NICs only and not any other NICs that use the same drivers.

2.2. DISCOVERING YOUR NUMA NODE TOPOLOGY

When you plan your deployment, you need to understand the NUMA topology of your Compute node to partition the CPU and memory resources for optimum performance. To determine the NUMA information, you can:

- Enable hardware introspection to retrieve this information from bare-metal nodes.
- Log onto each bare-metal node to manually collect the information.



NOTE

You must install and configure the undercloud before you can retrieve NUMA information through hardware introspection. See the [Director Installation and Usage Guide](#) for details.

Retrieving Hardware Introspection Details

The Bare Metal service hardware inspection extras (`inspection_extras`) is enabled by default to retrieve hardware details. You can use these hardware details to configure your overcloud. See [Configuring the Director](#) for details on the `inspection_extras` parameter in the `undercloud.conf` file.

For example, the `numa_topology` collector is part of these hardware inspection extras and includes the following information for each NUMA node:

- RAM (in kilobytes)
- Physical CPU cores and their sibling threads
- NICs associated with the NUMA node

Use the `openstack baremetal introspection data save _UUID_ | jq .numa_topology` command to retrieve this information, with the `UUID` of the bare-metal node.

The following example shows the retrieved NUMA information for a bare-metal node:

```
{
  "cpus": [
    {
      "cpu": 1,
      "thread_siblings": [
        1,
        17
      ],
      "numa_node": 0
    },
    {
      "cpu": 2,
      "thread_siblings": [
        10,
        26
      ],
      "numa_node": 1
    },
    {
      "cpu": 0,
      "thread_siblings": [
        0,
        16
      ],
      "numa_node": 0
    },
    {
      "cpu": 5,
      "thread_siblings": [
        13,
        29
      ],
      "numa_node": 1
    },
    {
      "cpu": 7,
```

```

    "thread_siblings": [
        15,
        31
    ],
    "numa_node": 1
},
{
    "cpu": 7,
    "thread_siblings": [
        7,
        23
    ],
    "numa_node": 0
},
{
    "cpu": 1,
    "thread_siblings": [
        9,
        25
    ],
    "numa_node": 1
},
{
    "cpu": 6,
    "thread_siblings": [
        6,
        22
    ],
    "numa_node": 0
},
{
    "cpu": 3,
    "thread_siblings": [
        11,
        27
    ],
    "numa_node": 1
},
{
    "cpu": 5,
    "thread_siblings": [
        5,
        21
    ],
    "numa_node": 0
},
{
    "cpu": 4,
    "thread_siblings": [
        12,
        28
    ],
    "numa_node": 1
},
{
    "cpu": 4,

```

```

        "thread_siblings": [
            4,
            20
        ],
        "numa_node": 0
    },
    {
        "cpu": 0,
        "thread_siblings": [
            8,
            24
        ],
        "numa_node": 1
    },
    {
        "cpu": 6,
        "thread_siblings": [
            14,
            30
        ],
        "numa_node": 1
    },
    {
        "cpu": 3,
        "thread_siblings": [
            3,
            19
        ],
        "numa_node": 0
    },
    {
        "cpu": 2,
        "thread_siblings": [
            2,
            18
        ],
        "numa_node": 0
    }
],
"ram": [
    {
        "size_kb": 66980172,
        "numa_node": 0
    },
    {
        "size_kb": 67108864,
        "numa_node": 1
    }
],
"nics": [
    {
        "name": "ens3f1",
        "numa_node": 1
    },
    {
        "name": "ens3f0",

```

```

        "numa_node": 1
    },
    {
        "name": "ens2f0",
        "numa_node": 0
    },
    {
        "name": "ens2f1",
        "numa_node": 0
    },
    {
        "name": "ens1f1",
        "numa_node": 0
    },
    {
        "name": "ens1f0",
        "numa_node": 0
    },
    {
        "name": "eno4",
        "numa_node": 0
    },
    {
        "name": "eno1",
        "numa_node": 0
    },
    {
        "name": "eno3",
        "numa_node": 0
    },
    {
        "name": "eno2",
        "numa_node": 0
    }
    ]
}

```

2.3. REVIEW BIOS SETTINGS

The following listing describes the required BIOS settings for NFV:

- **C3 Power State** - Disabled.
- **C6 Power State** - Disabled.
- **MLC Streamer** - Enabled.
- **MLC Spacial Prefetcher** - Enabled.
- **DCU Data Prefetcher** - Enabled.
- **DCA** - Enabled.
- **CPU Power and Performance** - Performance.

- **Memory RAS and Performance Config** → **NUMA Optimized** - Enabled.
- **Turbo Boost** - Disabled.

CHAPTER 3. SOFTWARE REQUIREMENTS

This section describes the supported configurations and drivers, and subscription details necessary for NFV.

To install Red Hat OpenStack Platform, you must register all systems in the OpenStack environment using the Red Hat Subscription Manager and subscribe to the required channels. See [Registering your system](#) for details.

3.1. SUPPORTED CONFIGURATIONS FOR NFV DEPLOYMENTS

Red Hat OpenStack Platform supports NFV deployments for SR-IOV and OVS-DPDK installation using the director. Using the composable roles feature available in the Red Hat OpenStack Platform director, you can create custom deployment roles. Hyper-converged Infrastructure (HCI), available with limited support for this release, allows you to colocate the Compute node with Red Hat Ceph Storage nodes for distributed NFV. To increase the performance in HCI, CPU pinning is used. The HCI model allows more efficient management in the NFV use cases. This release also provides OpenDaylight and Real-Time KVM as technology preview features. OpenDaylight is an open source modular, multi-protocol controller for Software-Defined Network (SDN) deployments. For more information on the support scope for features marked as technology previews, see [Technology Preview](#)

3.2. SUPPORTED DRIVERS

For a complete list of supported drivers, see [Component, Plug-In, and Driver Support in Red Hat OpenStack Platform](#).

For a list of NICs tested for NFV deployments with Red Hat OpenStack, see [Tested NICs](#).

3.3. COMPATIBILITY WITH THIRD PARTY SOFTWARE

For a complete list of products and services tested, supported, and certified to perform with Red Hat technologies (Red Hat OpenStack Platform), see [Third Party Software compatible with Red Hat OpenStack Platform](#). You can filter the list by product version and software category.

For a complete list of products and services tested, supported, and certified to perform with Red Hat technologies (Red Hat Enterprise Linux), see [Third Party Software compatible with Red Hat Enterprise Linux](#). You can filter the list by product version and software category.

CHAPTER 4. NETWORK CONSIDERATIONS

The undercloud host requires at least the following networks:

- Provisioning network - Provides DHCP and PXE boot functions to help discover bare-metal systems for use in the overcloud.
- External network - A separate network for remote connectivity to all nodes. The interface connecting to this network requires a routable IP address, either defined statically, or dynamically through an external DHCP service.

The minimal overcloud network configuration includes:

- Single NIC configuration - One NIC for the Provisioning network on the native VLAN and tagged VLANs that use subnets for the different overcloud network types.
- Dual NIC configuration - One NIC for the Provisioning network and the other NIC for the External network.
- Dual NIC configuration - One NIC for the Provisioning network on the native VLAN and the other NIC for tagged VLANs that use subnets for the different overcloud network types.
- Multiple NIC configuration - Each NIC uses a subnet for a different overcloud network type.

For more information on the networking requirements, see [Networking requirements](#).

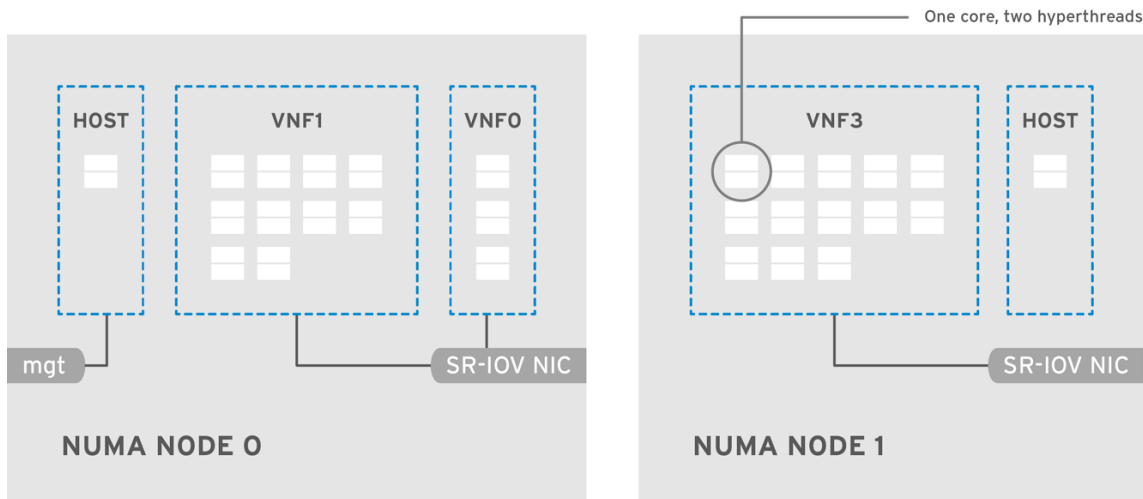
CHAPTER 5. PLANNING AN SR-IOV DEPLOYMENT

To optimize your SR-IOV deployment for NFV, you should understand how to set the individual OVS-DPDK parameters based on your Compute node hardware.

See [Discovering your NUMA node topology](#) to evaluate your hardware impact on the SR-IOV parameters.

5.1. HARDWARE PARTITIONING FOR AN SR-IOV DEPLOYMENT

To achieve high performance with SR-IOV, you need to partition the resources between the host and the guest.

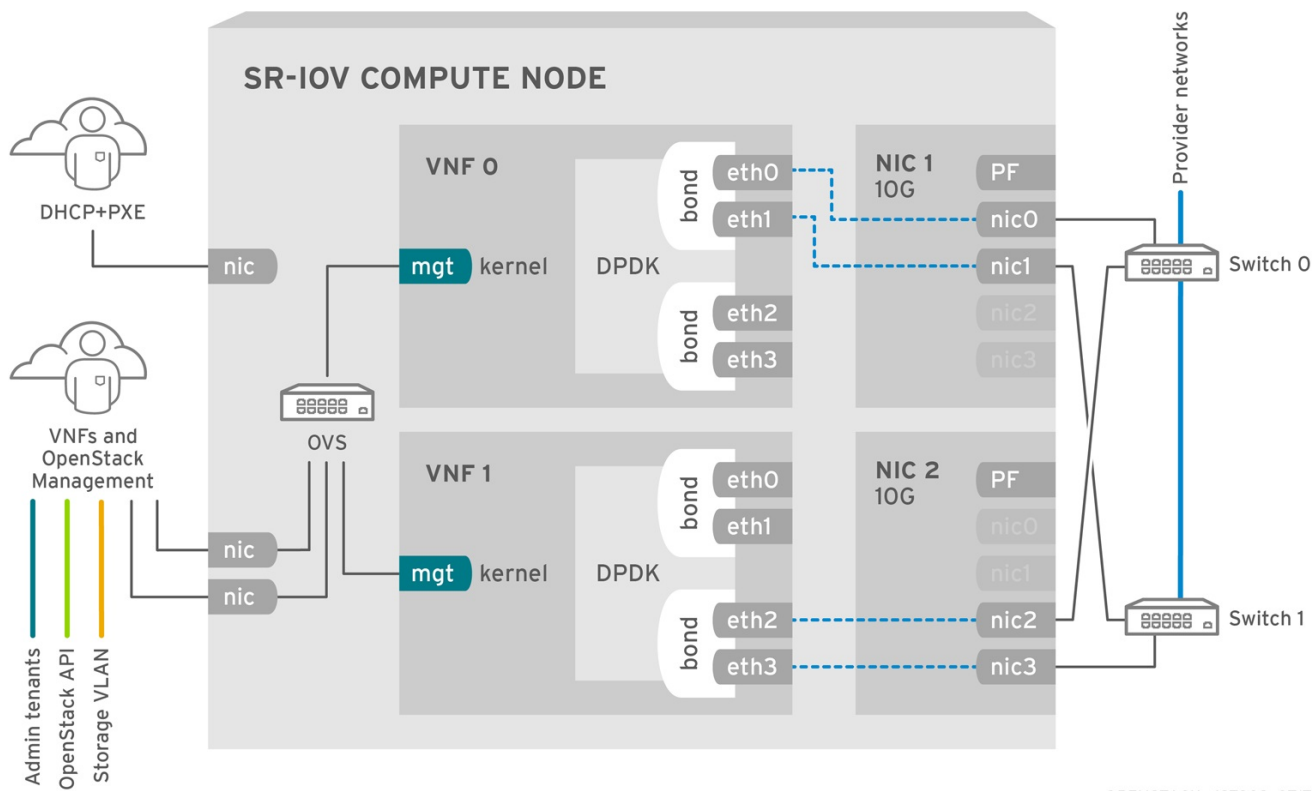


OPENSTACK_464931_0118

A typical topology includes 14 cores per NUMA node on dual socket Compute nodes. Both hyper-threading (HT) and non-HT cores are supported. Each core has two sibling threads. One core is dedicated to the host on each NUMA node. The VNF handles the SR-IOV interface bonding. All the interrupt requests (IRQs) are routed on the host cores. The VNF cores are dedicated to the VNFs. They provide isolation from other VNFs as well as isolation from the host. Each VNF must use resources on a single NUMA node. The SR-IOV NICs used by the VNF must also be associated with that same NUMA node. This topology does not have a virtualization overhead. The host, OpenStack Networking (neutron) and Compute (nova) configuration parameters are exposed in a single file for ease, consistency and to avoid incoherence that is fatal to proper isolation, causing preemption and packet loss. The host and virtual machine isolation depend on a tuned profile, which takes care of the boot parameters and any OpenStack modifications based on the list of CPUs to isolate.

5.2. TOPOLOGY OF AN NFV SR-IOV DEPLOYMENT

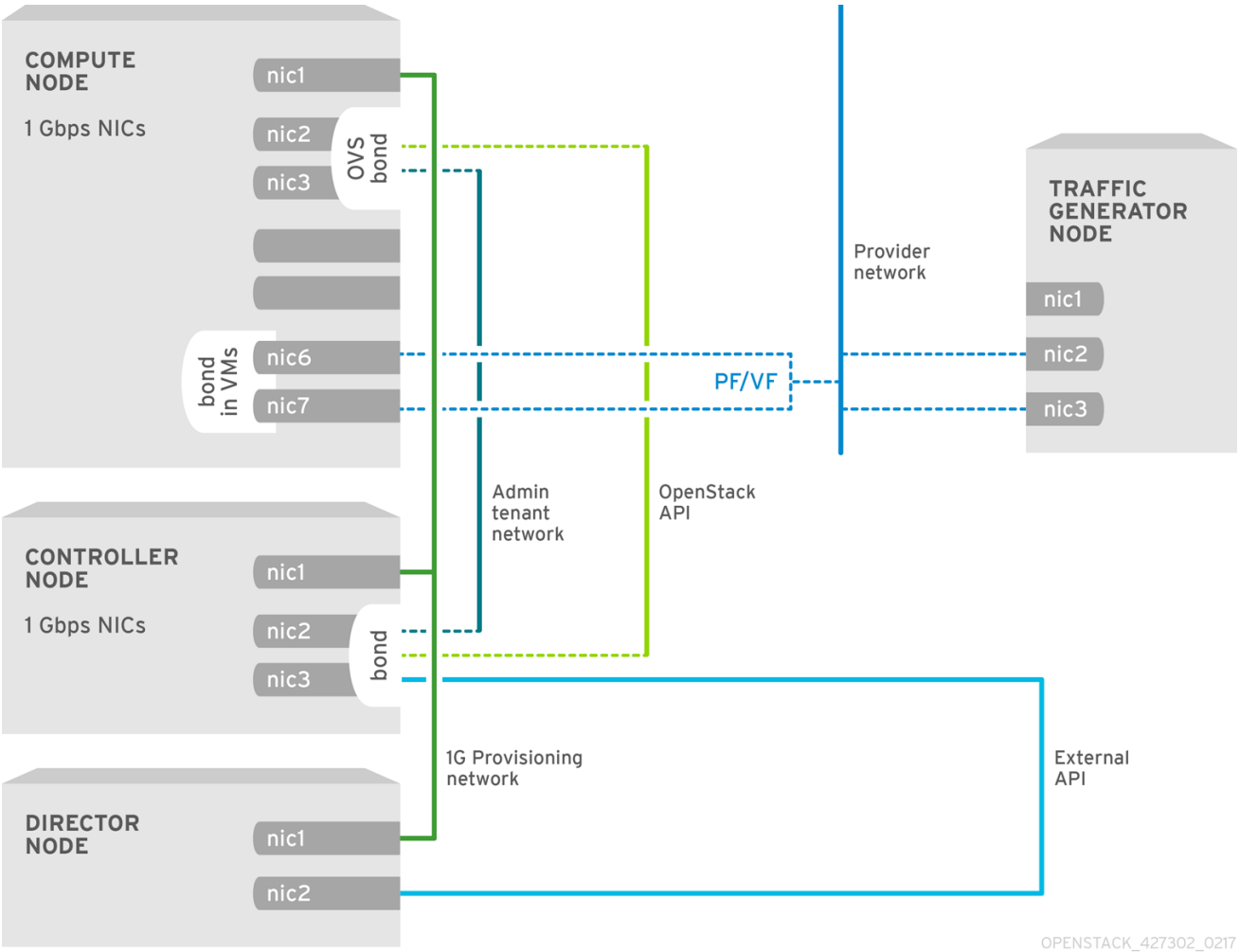
The following image has two VNFs each with the management interface represented by `mgt` and the data plane interfaces. The management interface manages the `ssh` access and so on. The data plane interfaces bond the VNFs to DPDK to ensure high availability (VNFs bond the data plane interfaces using the DPDK library). The image also has two redundant provider networks. The Compute node has two regular NICs bonded together and shared between the VNF management and the Red Hat OpenStack Platform API management.



The image shows a VNF that leverages DPDK at an application level and has access to SR-IOV VF/PFs, together for better availability or performance (depending on the fabric configuration). DPDK improves performance, while the VF/PF DPDK bonds provide support for failover (availability). The VNF vendor must ensure their DPDK PMD driver supports the SR-IOV card that is being exposed as a VF/PF. The management network uses OVS so the VNF sees a “mgt” network device using the standard virtIO drivers. Operators can use that device to initially connect to the VNF and ensure that their DPDK application bonds properly the two VF/PFs.

5.2.1. NFV SR-IOV without HCI

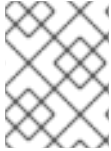
The following image shows the topology for SR-IOV without HCI for the NFV use case. It consists of Compute and Controller nodes with 1 Gbps NICs, and the Director node.



CHAPTER 6. CONFIGURING AN SR-IOV DEPLOYMENT

This section describes how to configure Single Root Input/Output Virtualization (SR-IOV) for Red Hat OpenStack.

You must install and configure the undercloud before you can deploy the overcloud. See the [Director Installation and Usage Guide](#) for details.

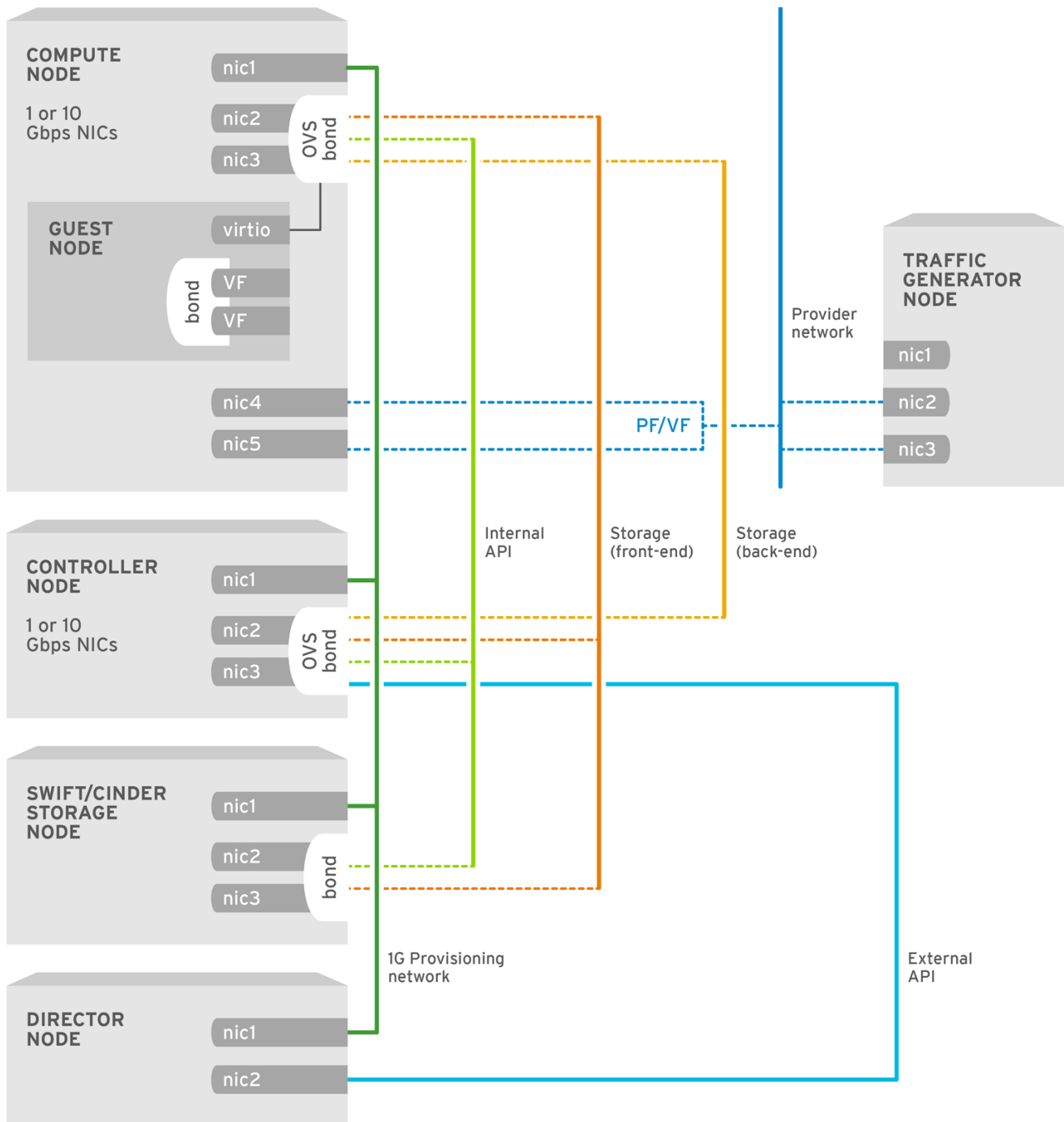


NOTE

Do not edit or change `isolated_cores` or other values in `etc/tuned/cpu-partitioning-variables.conf` that are modified by these director heat templates

6.1. OVERVIEW OF SR-IOV CONFIGURABLE PARAMETERS

You need to update the `network-environment.yaml` file to include parameters for kernel arguments, SR-IOV driver, PCI passthrough and so on. You must also update the `compute.yaml` file to include the SR-IOV interface parameters, and run the `overcloud_deploy.sh` script to deploy the overcloud with the SR-IOV parameters.



OPENSTACK_450694_0617

**NOTE**

This guide provides examples for CPU assignments, memory allocation, and NIC configurations that may vary from your topology and use case.

6.2. CONFIGURING SR-IOV WITH CONTROL PLANE BONDING

This section describes how to configure a composable role for SR-IOV with two port control plane bonding for your OpenStack environment. The process to create and deploy a composable role includes:

- Define the new role in a local copy of the `role_data.yaml` file.
- Modify the `network_environment.yaml` file to include this new role.

- Deploy the overcloud with this updated set of roles.

In this example, **ComputeSriov** is a composable role for compute node to enable SR-IOV only on the nodes that have the SR-IOV NICs. The existing set of default roles provided by the Red Hat OpenStack Platform is stored in the `/home/stack/roles_data.yaml` file.

6.2.1. Creating the ComputeSriov composable role

Modify `roles_data.yaml` to Create an SR-IOV Composable Role.

Copy the `roles_data.yaml` file to your `/home/stack/templates` directory and add the new **ComputeSriov** role.

```
- name: ComputeSriov
  description: |
    Compute Sriov Node role
  CountDefault: 1
  networks:
    - InternalApi
    - Tenant
    - Storage
  HostnameFormatDefault: computesriov-%index%
  ServicesDefault:
    - OS::TripleO::Services::AuditD
    - OS::TripleO::Services::CACerts
    - OS::TripleO::Services::CephClient
    - OS::TripleO::Services::CephExternal
    - OS::TripleO::Services::CertmongerUser
    - OS::TripleO::Services::Collectd
    - OS::TripleO::Services::ComputeCeilometerAgent
    - OS::TripleO::Services::ComputeNeutronCorePlugin
    - OS::TripleO::Services::ComputeNeutronL3Agent
    - OS::TripleO::Services::ComputeNeutronMetadataAgent
    - OS::TripleO::Services::ComputeNeutronOvsAgent
    - OS::TripleO::Services::Docker
    - OS::TripleO::Services::FluentdClient
    - OS::TripleO::Services::Iscsid
    - OS::TripleO::Services::Kernel
    - OS::TripleO::Services::MySQLClient
    - OS::TripleO::Services::NeutronLinuxbridgeAgent
    - OS::TripleO::Services::NeutronSriovAgent
    - OS::TripleO::Services::NeutronSriovHostConfig
    - OS::TripleO::Services::NeutronVppAgent
    - OS::TripleO::Services::NovaCompute
    - OS::TripleO::Services::NovaLibvirt
    - OS::TripleO::Services::NovaMigrationTarget
    - OS::TripleO::Services::Ntp
    - OS::TripleO::Services::ContainersLogrotateCronD
    - OS::TripleO::Services::OpenDaylightOvs
    - OS::TripleO::Services::Securetty
    - OS::TripleO::Services::SensuClient
    - OS::TripleO::Services::Snmp
    - OS::TripleO::Services::Sshd
    - OS::TripleO::Services::Timezone
    - OS::TripleO::Services::TripleoFirewall
    - OS::TripleO::Services::TripleoPackages
```


- OS::Triple0::Services::Tuned
- OS::Triple0::Services::Vpp
- OS::Triple0::Services::OVNController

6.2.2. Configuring tuned for CPU affinity

1. Create or modify [post-install.yaml](#) to set the tuned configuration to enable CPU affinity.

```
resources:

  ExtraDeployments:
    type: OS::Heat::StructuredDeployments
    properties:
      servers: {get_param: servers}
      config: {get_resource: ExtraConfig}
      actions: ['CREATE', 'UPDATE']

  ExtraConfig:
    type: OS::Heat::SoftwareConfig
    properties:
      group: script
      config: |
        #!/bin/bash
        set -x
        function tuned_service_dependency() {
          tuned_service=/usr/lib/systemd/system/tuned.service
          grep -q "network.target" $tuned_service
          if [ "$?" -eq 0 ]; then
            sed -i '/After=.*s/network.target//g'
            $tuned_service
          fi
          grep -q "Before=.*network.target" $tuned_service
          if [ ! "$?" -eq 0 ]; then
            grep -q "Before=.*" $tuned_service
            if [ "$?" -eq 0 ]; then
              sed -i 's/^\(Before=.*\)\/\1 network.target
openvswitch.service/g' $tuned_service
            else
              sed -i '/After/i Before=network.target
openvswitch.service' $tuned_service
            fi
          fi
        }

        if hiera -c /etc/puppet/hiera.yaml service_names | grep -q
neutron_ovs_dpdk_agent; then
          tuned_service_dependency
        fi
```

6.2.3. Configuring SR-IOV parameters

You must set the SR-IOV parameters to match your OpenStack deployment.

This example uses the sample [network-environment.yaml](#).

1. Add the custom resources for SR-IOV including the **ComputeSriov** role under **resource_registry**.

```
resource_registry:
    # Specify the relative/absolute path to the config files you
    # want to use for override the default.
    OS::TripleO::ComputeSriov::Net::SoftwareConfig: nic-
    configs/compute-sriov.yaml
    OS::TripleO::Controller::Net::SoftwareConfig: nic-
    configs/controller.yaml
    OS::TripleO::NodeExtraConfigPost: post-install.yaml
```

2. Under **parameter_defaults**, disable the tunnel type (set the value to `""`), and set network type to **vlan**.

```
NeutronTunnelTypes: ''
NeutronNetworkType: 'vlan'
```

3. **Optional:** Under **parameter_defaults**, map the physical network to the logical bridge. You will need to consider your OVS/OVS-DPSK configuration when determining whether your deployment requires this step.

```
NeutronBridgeMappings: 'tenant:br-link0'
```

4. Under **parameter_defaults**, set the OpenStack Networking ML2 and Open vSwitch VLAN mapping range.

```
NeutronNetworkVLANRanges: 'tenant:22:22,tenant:25:25'
```

This example sets the VLAN ranges on the physical network (**tenant**).

5. Under **parameter_defaults**, set the SR-IOV configuration parameters.

- a. Enable the SR-IOV mechanism driver (**sriovnicswitch**).

```
NeutronMechanismDrivers: "openvswitch,sriovnicswitch"
```

- b. Configure the Compute **pci_passthrough_whitelist** parameter, and set **devname** for the SR-IOV interface. The whitelist sets the PCI devices available to instances.

```
NovaPCIPassthrough:
    - devname: "ens2f0"
      physical_network: "tenant"
    - devname: "ens2f1"
      physical_network: "tenant"
```

- c. Specify the physical network and SR-IOV interface in the format - **PHYSICAL_NETWORK:PHYSICAL_DEVICE**.

All physical networks listed in the **network_vlan_ranges** on the server should have mappings to the appropriate interfaces on each agent.

```
NeutronPhysicalDevMappings: "tenant:ens2f0,tenant:ens2f1"
```

This example uses `tenant` as the `physical_network` name.

- d. Provide the number of Virtual Functions (VFs) to be reserved for each SR-IOV interface.

```
NeutronSriovNumVFs: "ens2f0:5,ens2f1:5"
```

This example reserves 5 VFs for the SR-IOV interfaces.



NOTE

Red Hat OpenStack Platform supports the number of VFs supported by the NIC vendor. See [Deployment Limits for Red Hat OpenStack Platform](#) for other related details.

6. Under `parameter_defaults`, set the role-specific parameters for the `ComputeSriov` role.

```
# SR-IOV compute node.
ComputeSriovParameters:
  KernelArgs: default_hugepagesz=1GB hugepagesz=1G hugepages=32
  iommu=pt intel_iommu=on
  TunedProfileName: "cpu-partitioning"
  IsolCpusList:
    "1,2,3,4,5,6,7,9,10,17,18,19,20,21,22,23,11,12,13,14,15,25,26,27,28,
    29,30,31"
  NovaVcpuPinSet:
    ['1,2,3,4,5,6,7,9,10,17,18,19,20,21,22,23,11,12,13,14,15,25,26,27,28,
    29,30,31']
  NovaReservedHostMemory: 4096
```



NOTE

You should also add `hw:mem_page_size=1GB` to the flavor you associate with the SR-IOV instance.

7. Under `parameter_defaults`, set the Metadata parameters.

```
# DHCP provide metadata route to VM.
NeutronEnableIsolatedMetadata: true
# DHCP always provides metadata route to VM.
NeutronEnableForceMetadata: true
```

8. Under `parameter_defaults`, list the applicable filters.

Nova scheduler applies these filters in the order they are listed. List the most restrictive filters first to make the filtering process for the nodes more efficient.

```
NovaSchedulerDefaultFilters:
['AvailabilityZoneFilter', 'RamFilter', 'ComputeFilter', 'ComputeCapabi
litiesFilter', 'ImagePropertiesFilter', 'ServerGroupAntiAffinityFilter
', 'ServerGroupAffinityFilter', 'PciPassthroughFilter']
```

6.2.4. Configuring the Controller node

This example uses the sample [controller.yaml](#) file.

1. Create the interface for an isolated network.

```
- type: linux_bond
  name: bond_api
  bonding_options: "mode=active-backup"
  use_dhcp: false
  dns_servers:
    get_param: DnsServers
  addresses:
  - ip_netmask:
      list_join:
      - /
      - - get_param: ControlPlaneIp
        - get_param: ControlPlaneSubnetCidr
  routes:
  - ip_netmask: 169.254.169.254/32
    next_hop:
      get_param: EC2MetadataIp
  members:
  - type: interface
    name: nic2
    primary: true
```

2. Assign VLANs to this interface.

```
- type: vlan
  vlan_id:
    get_param: InternalApiNetworkVlanID
  device: bond_api
  addresses:
  - ip_netmask:
      get_param: InternalApiIpSubnet

- type: vlan
  vlan_id:
    get_param: TenantNetworkVlanID
  device: bond_api
  addresses:
  - ip_netmask:
      get_param: TenantIpSubnet

- type: vlan
  vlan_id:
    get_param: StorageNetworkVlanID
  device: bond_api
  addresses:
  - ip_netmask:
      get_param: StorageIpSubnet

- type: vlan
  vlan_id:
    get_param: StorageMgmtNetworkVlanID
  device: bond_api
  addresses:
```

```

- ip_netmask:
  get_param: StorageMgmtIpSubnet

- type: vlan
  vlan_id:
    get_param: ExternalNetworkVlanID
  device: bond_api
  addresses:
    - ip_netmask:
      get_param: ExternalIpSubnet
  routes:
    - default: true
      next_hop:
        get_param: ExternalInterfaceDefaultRoute

```

3. Create the OVS bridge for access to the floating IPs into cloud networks.

```

-
- type: ovs_bridge
  name: br-link0
  use_dhcp: false
  members:
    - type: interface
      name: nic3
      mtu: 9000

```

6.2.5. Configuring the Compute node for SR-IOV interfaces

This example uses the sample `compute-sriov.yaml` file.

Create `compute-sriov.yaml` from the default `compute.yaml` file. This is the file that controls the parameters for the Compute nodes that use the `ComputeSriov` composable role.

1. Create the interface for an isolated network.

```

- type: linux_bond
  name: bond_api
  bonding_options: "mode=active-backup"
  use_dhcp: false
  dns_servers:
    get_param: DnsServers
  members:
    - type: interface
      name: nic3
      primary: true
    - type: interface
      name: nic4

```

2. Assign VLANs to this interface.

```

- type: vlan
  vlan_id:
    get_param: InternalApiNetworkVlanID
  device: bond_api
  addresses:

```

- ip_netmask:
get_param: InternalApiIpSubnet
- type: vlan
vlan_id:
get_param: TenantNetworkVlanID
device: bond_api
addresses:
 - ip_netmask:
get_param: TenantIpSubnet
- type: vlan
vlan_id:
get_param: StorageNetworkVlanID
device: bond_api
addresses:
 - ip_netmask:
get_param: StorageIpSubnet

3. Create a interfaces to the Controller node.

- type: interface
name: ens2f0
mtu: 9000
use_dhcp: false
defroute: false
nm_controlled: true
hotplug: true
- type: interface
name: ens2f1
mtu: 9000
use_dhcp: false
defroute: false
nm_controlled: true
hotplug: true

6.2.6. Deploying the overcloud

Run the `overcloud_deploy.sh` script to deploy the overcloud.

```
#!/bin/bash

openstack overcloud deploy \
  --templates \
  -r /home/stack/ospd-12-vlan-sriov-two-ports-ctlplane-
bonding/roles_data.yaml \
  -e /usr/share/openstack-tripleo-heat-templates/environments/network-
isolation.yaml \
  -e /usr/share/openstack-tripleo-heat-templates/environments/neutron-
sriov.yaml \
  -e /usr/share/openstack-tripleo-heat-templates/environments/host-config-
and-reboot.yaml \
  -e /home/stack/ospd-12-vlan-sriov-two-ports-ctlplane-bonding/docker-
```

```
images.yaml \
-e /home/stack/ospd-12-vlan-sriov-two-ports-ctlplane-bonding/network-
environment.yaml
```

6.3. CREATING A FLAVOR AND DEPLOYING AN INSTANCE FOR SR-IOV

After you have completed configuring SR-IOV for your Red Hat OpenStack Platform deployment with NFV, you need to create a flavor and deploy an instance by performing the following steps:

1. Create an aggregate group and add a host to it for SR-IOV.

```
# openstack aggregate create --zone=sriov sriov
# openstack aggregate add host sriov compute-sriov-0.localdomain
```



NOTE

You should use host aggregates to separate CPU pinned instances from unpinned instances. Instances that do not use CPU pinning do not respect the resourcing requirements of instances that use CPU pinning.

2. Create a flavor.

```
# openstack flavor create compute --ram 4096 --disk 150 --vcpus 4
```

compute is the flavor name, **4096** is the memory size in MB, **150** is the disk size in GB (default 0G), and **4** is the number of vCPUs.

3. Set additional flavor properties.

```
# openstack flavor set --property hw:cpu_policy=dedicated --property
hw:mem_page_size=1GB compute
```

compute is the flavor name and the remaining parameters set the other properties for the flavor.

4. Create the network.

```
# openstack network create net1 --provider-physical-network tenant -
-provider-network-type vlan --provider-segment <VLAN-ID>
```

5. Create the port.

- a. Use **vnic-type direct** to create an SR-IOV VF port:

```
# openstack port create --network net1 --vnic-type direct
sriov_port
```

- b. Use **vnic-type direct-physical** to create an SR-IOV PF port.

```
# openstack port create --network net1 --vnic-type direct-
physical sriov_port
```

6. Deploy an instance.

```
# openstack server create --flavor compute --availability-zone sriov  
--image rhel_7.3 --nic port-id=sriov_port sriov_vm
```

Where:

- **compute** is the flavor name or ID.
- **sriov** is the availability zone for the server.
- **rhel_7.3** is the image (name or ID) used to create an instance.
- **sriov_port** is the NIC on the server.
- **sriov_vm** is the name of the instance.

You have now deployed an instance for the SR-IOV with NFV use case.

CHAPTER 7. PLANNING YOUR OVS-DPDK DEPLOYMENT

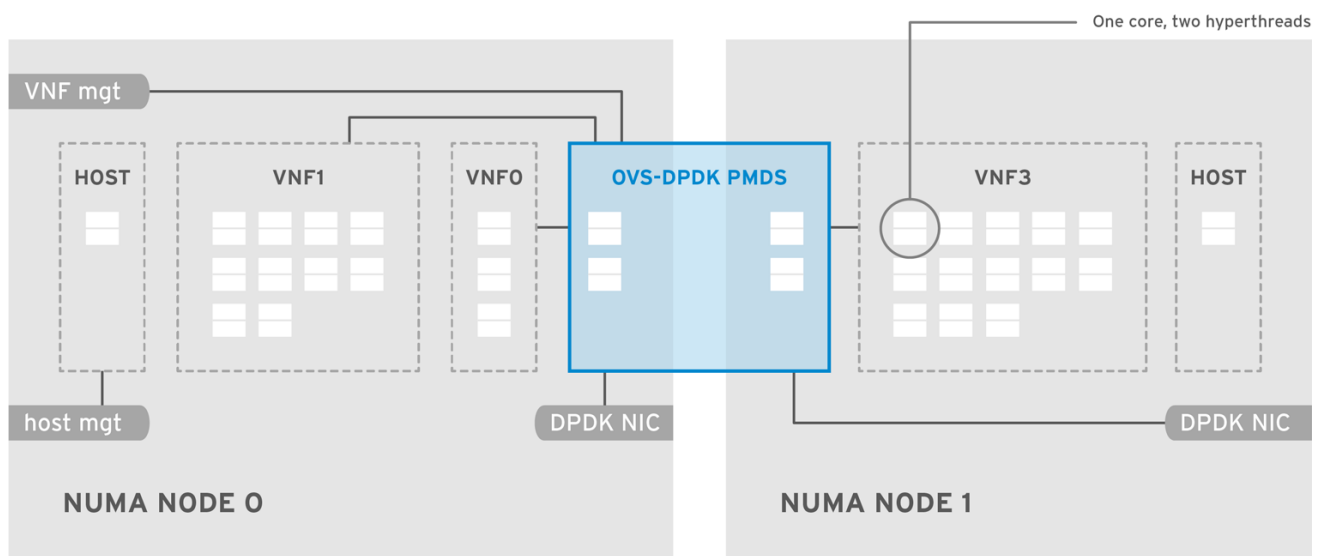
To optimize your OVS-DPDK deployment for NFV, you should understand how OVS-DPDK uses the Compute node hardware (CPU, NUMA nodes, memory, NICs) and the considerations for determining the individual OVS-DPDK parameters based on your Compute node.

See [NFV performance considerations](#) for a high-level introduction to CPUs and NUMA topology.

7.1. OVS-DPDK WITH CPU PARTITIONING AND NUMA TOPOLOGY

OVS-DPDK partitions the hardware resources for host, guests, and OVS-DPDK itself. The OVS-DPDK Poll Mode Drivers (PMDs) run DPDK active loops, which require dedicated cores. This means a list of CPUs and Huge Pages are dedicated to OVS-DPDK.

A sample partitioning includes 16 cores per NUMA node on dual socket Compute nodes. The traffic requires additional NICs since the NICs cannot be shared between the host and OVS-DPDK.



OPENSTACK_464931_0118



NOTE

DPDK PMD threads must be reserved on both NUMA nodes even if a NUMA node does not have an associated DPDK NIC.

OVS-DPDK performance also depends on reserving a block of memory local to the NUMA node. Use NICs associated with the same NUMA node that you use for memory and CPU pinning. Also ensure both interfaces in a bond are from NICs on the same NUMA node.

7.2. OVERVIEW OF WORKFLOWS AND DERIVED PARAMETERS



IMPORTANT

This feature is available in this release as a *Technology Preview*, and therefore is not fully supported by Red Hat. It should only be used for testing, and should not be deployed in a production environment. For more information about Technology Preview features, see [Scope of Coverage Details](#).

You can use the OpenStack Workflow (mistral) service to derive parameters based on the capabilities of your available bare-metal nodes. Openstack workflows use a `.yaml` file to define a set of tasks and actions to perform. You can use a pre-defined workbook, `derive_params.yaml`, in the `tripleo-common/workbooks/` directory. This workbook provides workflows to derive each supported parameter from the results retrieved from Bare Metal introspection. The `derive_params.yaml` workflows use the formulas from `tripleo-common/workbooks/derive_params_formulas.yaml` to calculate the derived parameters.



NOTE

You can modify the formulas in `derive_params_formulas.yaml` to suit your environment.

The `derive_params.yaml` workbook assumes all nodes for a *given composable* role have the same hardware specifications. The workflow considers the flavor-profile association and nova placement scheduler to match nodes associated with a role and uses the introspection data from the first node that matches the role.

See [Troubleshooting Workflows and Executions](#) for details on OpenStack workflows.

You can use the `-p` or `--plan-environment-file` option to add a custom `plan_environment.yaml` file to the `openstack overcloud deploy` command. The custom `plan_environment.yaml` file provides the list of workbooks and any input values to pass into the workbook. The triggered workflows merge the derived parameters back into the custom `plan_environment.yaml`, where they are available for the overcloud deployment. You can use these derived parameter results to prepare your overcloud images.

See [Plan Environment Metadata](#) for details on how to use the `--plan-environment-file` option in your deployment.

7.3. DERIVED OVS-DPDK PARAMETERS

The workflows in `derive_params.yaml` derive the DPDK parameters associated with the matching role that uses the `ComputeNeutronOvsDpdk` service.

The following is the list of parameters the workflows can automatically derive for OVS-DPDK:

- `IsolCpusList`
- `KernelArgs`
- `NovaReservedHostMemory`
- `NovaVcpuPinSet`
- `OvsDpdkCoreList`
- `OvsDpdkSocketMemory`
- `OvsPmdCoreList`

The `OvsDpdkMemoryChannels` parameter cannot be derived from the introspection memory bank data since the format of memory slot names are not consistent across different hardware environments.

In most cases, `OvsDpdkMemoryChannels` should be 4 (default). Use your hardware manual to determine the number of memory channels per socket and use this value to override the default.

See [Section 8.1, “Deriving DPDK parameters with workflows”](#) for configuration details.

7.4. OVERVIEW OF MANUALLY CALCULATED OVS-DPDK PARAMETERS

This section describes how OVS-DPDK uses parameters within the director `network_environment.yaml` HEAT templates to configure the CPU and memory for optimum performance. Use this information to evaluate the hardware support on your Compute nodes and how best to partition that hardware to optimize your OVS-DPDK deployment.



NOTE

You do not need to manually calculate these parameters if you use the `derived_parameters.yaml` workflow to generate these values automatically. See [Overview of workflows and derived parameters](#)



NOTE

Always pair CPU sibling threads (logical CPUs) together for the physical core when allocating CPU cores.

See [Discovering your NUMA node topology](#) to determine the CPU and NUMA nodes on your Compute nodes. You use this information to map CPU and other parameters to support the host, guest instance, and OVS-DPDK process needs.

7.4.1. CPU parameters

OVS-DPDK uses the following CPU partitioning parameters:

OvsPmdCoreList

Provides the CPU cores that are used for the DPDK poll mode drivers (PMD). Choose CPU cores that are associated with the local NUMA nodes of the DPDK interfaces. `OvsPmdCoreList` is used for the `pmd-cpu-mask` value in Open vSwitch.

- Pair the sibling threads together.
- Exclude all cores from the `OvsDpdkCoreList`
- Avoid allocating the logical CPUs (both thread siblings) of the first physical core on both NUMA nodes as these should be used for the `OvsDpdkCoreList` parameter.
- Performance depends on the number of physical cores allocated for this PMD Core list. On the NUMA node which is associated with DPDK NIC, allocate the required cores.
- For NUMA nodes with a DPDK NIC:
 - Determine the number of physical cores required based on the performance requirement and include all the sibling threads (logical CPUs) for each physical core.
- For NUMA nodes without DPDK NICs:

- o Allocate the sibling threads (logical CPUs) of one physical core (excluding the first physical core of the NUMA node). You need a minimal DPDK poll mode driver on the NUMA node even without DPDK NICs present to avoid failures in creating guest instances.

**NOTE**

DPDK PMD threads must be reserved on both NUMA nodes even if a NUMA node does not have an associated DPDK NIC.

NovaVcpuPinSet

Sets cores for CPU pinning. The Compute node uses these cores for guest instances. **NovaVcpuPinSet** is used as the `vcpu_pin_set` value in the `nova.conf` file.

- Exclude all cores from the **OvsPmdCoreList** and the **OvsDpdkCoreList**.
- Include all remaining cores.
- Pair the sibling threads together.

IsolCpusList

A set of CPU cores isolated from the host processes. This parameter is used as the `isolated_cores` value in the `cpu-partitioning-variable.conf` file for the `tuned-profiles-cpu-partitioning` component.

- Match the list of cores in **OvsPmdCoreList** and **NovaVcpuPinSet**.
- Pair the sibling threads together.

OvsDpdkCoreList

Provides CPU cores for non data path OVS-DPDK processes, such as handler and revalidator threads. This parameter has no impact on overall data path performance on multi-NUMA node hardware. This parameter is used for the `dpdk-1core-mask` value in Open vSwitch, and these cores are shared with the host.

- Allocate the first physical core (and sibling thread) from each NUMA node (even if the NUMA node has no associated DPDK NIC).
- These cores must be mutually exclusive from the list of cores in **OvsPmdCoreList** and **NovaVcpuPinSet**.

7.4.2. Memory parameters

OVS-DPDK uses the following memory parameters:

OvsDpdkMemoryChannels

Maps memory channels in the CPU per NUMA node. The **OvsDpdkMemoryChannels** parameter is used by Open vSwitch as the `other_config:dpdk-extra="-n <value>"` value.

- Use `dmidecode -t memory` or your hardware manual to determine the number of memory channels available.

- Use `ls /sys/devices/system/node/node* -d` to determine the number of NUMA nodes.
- Divide the number of memory channels available by the number of NUMA nodes.

NovaReservedHostMemory

Reserves memory in MB for tasks on the host. This value is used by the Compute node as the `reserved_host_memory_mb` value in `nova.conf`.

- Use the static recommended value of 4096 MB.

OvsDpdkSocketMemory

Specifies the amount of memory in MB to pre-allocate from the hugepage pool, per NUMA node. This value is used by Open vSwitch as the `other_config:dpdk-socket-mem` value.

- Provide as a comma-separated list. The `OvsDpdkSocketMemory` value is calculated from the MTU value of each NIC on the NUMA node.
- For a NUMA node without a DPDK NIC, use the static recommendation of 1024 MB (1GB)
- The following equation approximates the value for `OvsDpdkSocketMemory`:
 - $\text{MEMORY_REQD_PER_MTU} = (\text{ROUNDUP_PER_MTU} + 800) * (4096 * 64)$ Bytes
 - 800 is the overhead value.
 - $4096 * 64$ is the number of packets in the mempool.
- Add the `MEMORY_REQD_PER_MTU` for each of the MTU values set on the NUMA node and add another 512 MB as buffer. Round the value up to a multiple of 1024.

Sample Calculation - MTU 2000 and MTU 9000

DPDK NICs `dpdk0` and `dpdk1` are on the same NUMA node 0 and configured with MTUs 9000 and 2000 respectively. The sample calculation to derive the memory required is as follows:

1. Round off the MTU values to the nearest 1024 bytes.

The MTU value of 9000 becomes 9216 bytes.
The MTU value of 2000 becomes 2048 bytes.

2. Calculate the required memory for each MTU value based on these rounded byte values.

Memory required for 9000 MTU = $(9216 + 800) * (4096 * 64) = 2625634304$
Memory required for 2000 MTU = $(2048 + 800) * (4096 * 64) = 746586112$

3. Calculate the combined total memory required, in bytes.

$2625634304 + 746586112 + 536870912 = 3909091328$ bytes.

This calculation represents (Memory required for MTU of 9000) + (Memory required for MTU of 2000) + (512 MB buffer).

4. Convert the total memory required into MB.

```
3909091328 / (1024*1024) = 3728 MB.
```

5. Round this value up to the nearest 1024.

```
3724 MB rounds up to 4096 MB.
```

6. Use this value to set `OvsDpdkSocketMemory`.

```
OvsDpdkSocketMemory: "4096,1024"
```

Sample Calculation - MTU 2000

DPDK NICs `dpdk0` and `dpdk1` are on the same NUMA node 0 and configured with MTUs 2000 and 2000 respectively. The sample calculation to derive the memory required is as follows:

1. Round off the MTU values to the nearest 1024 bytes.

```
The MTU value of 2000 becomes 2048 bytes.
```

2. Calculate the required memory for each MTU value based on these rounded byte values.

```
Memory required for 2000 MTU = (2048 + 800) * (4096*64) = 746586112
```

3. Calculate the combined total memory required, in bytes.

```
746586112 + 536870912 = 1283457024 bytes.
```

This calculation represents (Memory required for MTU of 2000) + (512 MB buffer).

4. Convert the total memory required into MB.

```
1283457024 / (1024*1024) = 1224 MB.
```

5. Round this value up to the nearest 1024.

```
1224 MB rounds up to 2048 MB.
```

6. Use this value to set `OvsDpdkSocketMemory`.

```
OvsDpdkSocketMemory: "2048,1024"
```

7.4.3. Networking parameters

NeutronDpdkDriverType

Sets the driver type used by DPDK. Use the default of `vfio-pci`.

NeutronDatapathType

Datapath type for OVS bridges. DPDK uses the default value of `netdev`.

NeutronVhostuserSocketDir

Sets the vhost-user socket directory for OVS. Use `/var/lib/vhost_sockets` for vhost client mode.

7.4.4. Other parameters

NovaSchedulerDefaultFilters

Provides an ordered list of filters that the Compute node uses to find a matching Compute node for a requested guest instance.

KernelArgs

Provides multiple kernel arguments to `/etc/default/grub` for the Compute node at boot time. Add the following based on your configuration:

- **hugepagesz**: Sets the size of the huge pages on a CPU. This value can vary depending on the CPU hardware. Set to 1G for OVS-DPDK deployments (**default_hugepagesz=1GB** **hugepagesz=1G**). Check for the **pdpe1gb** CPU flag to ensure your CPU supports 1G.

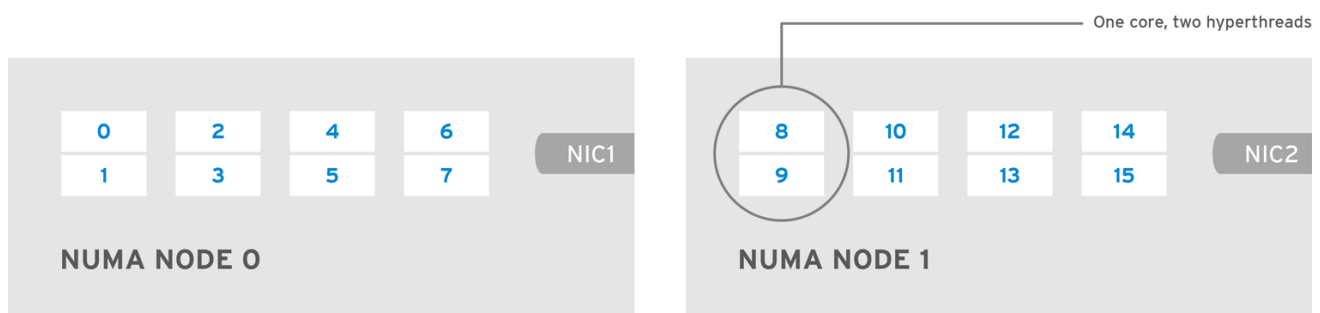
```
lshw -class processor | grep pdpe1gb
```

- **hugepages count**: Sets the number of huge pages available. This value depends on the amount of host memory available. Use most of your available memory (excluding **NovaReservedHostMemory**). You must also configure the huge pages count value within the OpenStack flavor associated with your Compute nodes.
- **iommu**: For Intel CPUs, add `"intel_iommu=on iommu=pt"`
- **isolcpus**: Sets the CPU cores to be tuned. This value matches **IsolCpusList**.

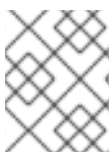
7.5. TWO NUMA NODE EXAMPLE OVS-DPDK DEPLOYMENT

This sample Compute node includes two NUMA nodes as follows:

- NUMA 0 has cores 0-7. The sibling thread pairs are (0,1), (2,3), (4,5), and (6,7)
- NUMA 1 has cores 8-15. The sibling thread pairs are (8,9), (10,11), (12,13), and (14,15).
- Each NUMA node connects to a physical NIC (NIC1 on NUMA 0 and NIC2 on NUMA 1).



OPENSTACK_453316_0717



NOTE

Reserve the first physical cores (both thread pairs) on each NUMA node (0,1 and 8,9) for non data path DPDK processes (**OvsDpdkCoreList**).

This example also assumes a 1500 MTU configuration, so the **OvsDpdkSocketMemory** is the same for all use cases:

```
OvsDpdkSocketMemory: "1024, 1024"
```

NIC 1 for DPDK, with one physical core for PMD

In this use case, you allocate one physical core on NUMA 0 for PMD. You must also allocate one physical core on NUMA 1, even though there is no DPDK enabled on the NIC for that NUMA node. The remaining cores (not reserved for `OvsDpdkCoreList`) are allocated for guest instances. The resulting parameter settings are:

```
OvsPmdCoreList: "2, 3, 10, 11"
NovaVcpuPinSet: "4, 5, 6, 7, 12, 13, 14, 15"
```

NIC 1 for DPDK, with two physical cores for PMD

In this use case, you allocate two physical cores on NUMA 0 for PMD. You must also allocate one physical core on NUMA 1, even though there is no DPDK enabled on the NIC for that NUMA node. The remaining cores (not reserved for `OvsDpdkCoreList`) are allocated for guest instances. The resulting parameter settings are:

```
OvsPmdCoreList: "2, 3, 4, 5, 10, 11"
NovaVcpuPinSet: "6, 7, 12, 13, 14, 15"
```

NIC 2 for DPDK, with one physical core for PMD

In this use case, you allocate one physical core on NUMA 1 for PMD. You must also allocate one physical core on NUMA 0, even though there is no DPDK enabled on the NIC for that NUMA node. The remaining cores (not reserved for `OvsDpdkCoreList`) are allocated for guest instances. The resulting parameter settings are:

```
OvsPmdCoreList: "2, 3, 10, 11"
NovaVcpuPinSet: "4, 5, 6, 7, 12, 13, 14, 15"
```

NIC 2 for DPDK, with two physical cores for PMD

In this use case, you allocate two physical cores on NUMA 1 for PMD. You must also allocate one physical core on NUMA 0, even though there is no DPDK enabled on the NIC for that NUMA node. The remaining cores (not reserved for `OvsDpdkCoreList`) are allocated for guest instances. The resulting parameter settings are:

```
OvsPmdCoreList: "2, 3, 10, 11, 12, 13"
NovaVcpuPinSet: "4, 5, 6, 7, 14, 15"
```

NIC 1 and NIC2 for DPDK, with two physical cores for PMD

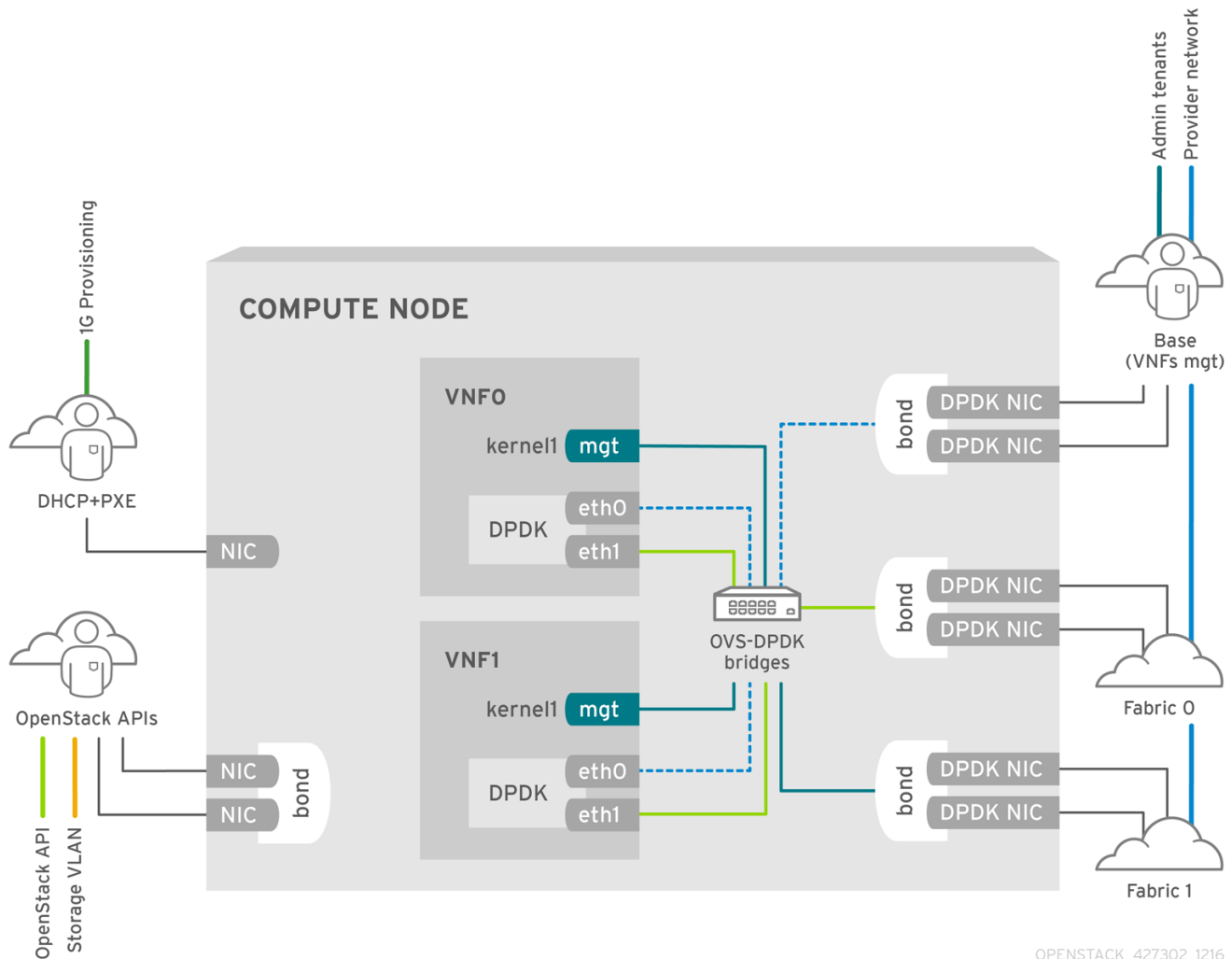
In this use case, you allocate two physical cores on each NUMA node for PMD. The remaining cores (not reserved for `OvsDpdkCoreList`) are allocated for guest instances. The resulting parameter settings are:

```
OvsPmdCoreList: "2, 3, 4, 5, 10, 11, 12, 13"
NovaVcpuPinSet: "6, 7, 14, 15"
```

7.6. TOPOLOGY OF AN NFV OVS-DPDK DEPLOYMENT

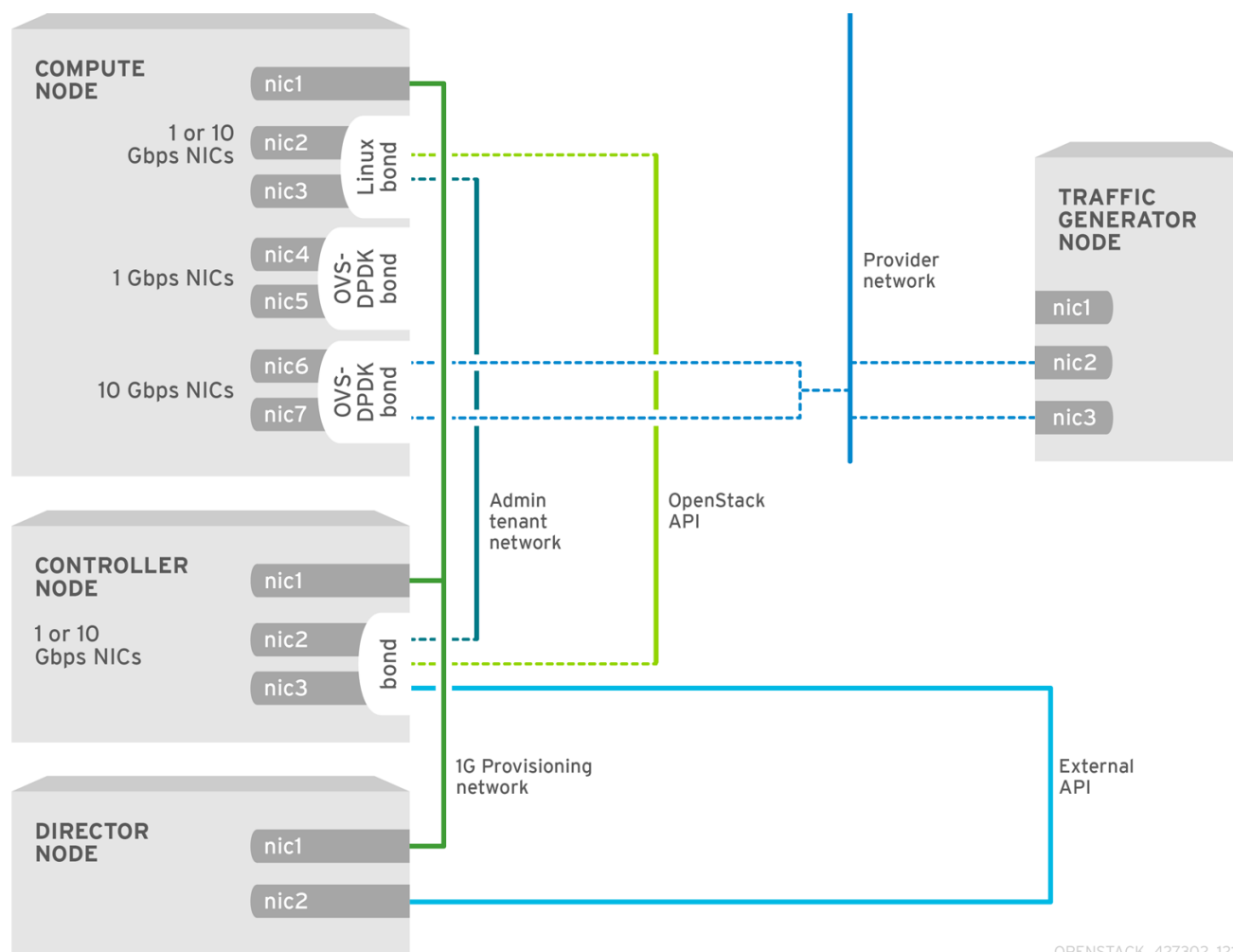
This sample OVS-DPDK deployment consists of two VNFs each with two interfaces, namely, the

management interface represented by `mgt` and the data plane interface. In the OVS-DPDK deployment, the VNFs run with inbuilt DPDK that supports the physical interface. OVS-DPDK takes care of the bonding at the vSwitch level. In an OVS-DPDK deployment, it is recommended that you **do not** mix kernel and OVS-DPDK NICs as it can lead to performance degradation. To separate the management (`mgt`) network, connected to the Base provider network for the virtual machine, you need to ensure you have additional NICs. The Compute node consists of two regular NICs for the OpenStack API management that can be reused by the Ceph API but cannot be shared with any OpenStack tenant.



NFV OVS-DPDK topology

The following image shows the topology for OVS_DPDK for the NFV use case. It consists of Compute and Controller nodes with 1 or 10 Gbps NICs, and the Director node.



CHAPTER 8. CONFIGURING AN OVS-DPDK DEPLOYMENT

This section deploys DPDK with Open vSwitch (OVS-DPDK) within the Red Hat OpenStack Platform environment. The overcloud usually consists of nodes in predefined roles such as Controller nodes, Compute nodes, and different storage node types. Each of these default roles contains a set of services defined in the core Heat templates on the director node.

You must install and configure the undercloud before you can deploy the overcloud. See the [Director Installation and Usage Guide](#) for details.



IMPORTANT

You must determine the best values for the OVS-DPDK parameters that you set in the `network-environment.yaml` file to optimize your OpenStack network for OVS-DPDK.



NOTE

Do not edit or change `isolated_cores` or other values in `etc/tuned/cpu-partitioning-variables.conf` that are modified by these director heat templates

8.1. DERIVING DPDK PARAMETERS WITH WORKFLOWS



IMPORTANT

This feature is available in this release as a *Technology Preview*, and therefore is not fully supported by Red Hat. It should only be used for testing, and should not be deployed in a production environment. For more information about Technology Preview features, see [Scope of Coverage Details](#).

See [Section 7.2, “Overview of workflows and derived parameters”](#) for an overview of the Mistral workflow for DPDK.

Prerequisites

You must have Bare Metal introspection, including hardware inspection extras (`inspection_extras`) enabled to provide the data retrieved by this workflow. Hardware inspection extras are enabled by default. See [Inspecting the Hardware of Nodes](#).

Define the Workflows and Input Parameters for DPDK

The following lists the input parameters you can provide to the OVS-DPDK workflows:

`num_phy_cores_per_numa_node_for_pmd`

This input parameter specifies the required minimum number of cores for the NUMA node associated with the DPDK NIC. One physical core is assigned for the other NUMA nodes not associated with DPDK NIC. This parameter should be set to 1.

`huge_page_allocation_percentage`

This input parameter specifies the required percentage of total memory (excluding `NovaReservedHostMemory`) that can be configured as huge pages. The `KernelArgs` parameter is derived using the calculated huge pages based on the `huge_page_allocation_percentage` specified. This parameter should be set to 50.

The workflows use these input parameters along with the bare-metal introspection details to calculate appropriate DPDK parameter values.

To define the workflows and input parameters for DPDK:

1. Copy the `tripleo-heat-templates/plan-samples/plan-environment-derived-params.yaml` file to a local directory and set the input parameters to suit your environment.

```
workflow_parameters:
  tripleo.derive_params.v1.derive_parameters:
    # DPDK Parameters #
    # Specifies the minimum number of CPU physical cores to be
    allocated for DPDK
    # PMD threads. The actual allocation will be based on network
    config, if
    # the a DPDK port is associated with a numa node, then this
    configuration
    # will be used, else 1.
    num_phy_cores_per_numa_node_for_pmd: 1
    # Amount of memory to be configured as huge pages in
    percentage. Out of the
    # total available memory (excluding the
    NovaReservedHostMemory), the
    # specified percentage of the remaining is configured as huge
    pages.
    huge_page_allocation_percentage: 50
```

2. Deploy the overcloud with the `update-plan-only` parameter to calculate the derived parameters.

```
$ openstack overcloud deploy --templates --update-plan-only -r
/home/stack/ospd-12-sriov-dpdk-heterogeneous-cluster/roles_data.yaml
-e /usr/share/openstack-tripleo-heat-templates/environments/network-
isolation.yaml -e /home/stack/ospd-12-sriov-dpdk-heterogeneous-
cluster/docker-images.yaml -e /usr/share/openstack-tripleo-heat-
templates/environments/host-config-and-reboot.yaml -e
/home/stack/ospd-12-sriov-dpdk-heterogeneous-cluster/network-
environment.yaml -e /usr/share/openstack-tripleo-heat-
templates/environments/neutron-sriov.yaml
-e /usr/share/openstack-tripleo-heat-
templates/environments/neutron-ovs-dpdk.yaml
-p /home/stack/plan-environment-derived-params.yaml
```

The output of this command shows the derived results, which are also merged into the `plan-environment.yaml` file.

```
Started Mistral Workflow
tripleo.validations.v1.check_pre_deployment_validations. Execution ID:
55ba73f2-2ef4-4da1-94e9-eae2fdc35535
Waiting for messages on queue 472a4180-e91b-4f9e-bd4c-1fbdfbcf414f with no
timeout.
Removing the current plan files
Uploading new plan files
Started Mistral Workflow
tripleo.plan_management.v1.update_deployment_plan. Execution ID: 7fa995f3-
```

```
7e0f-4c9e-9234-dd5292e8c722
```

```
Plan updated.
```

```
Processing templates in the directory /tmp/tripleoclient-SY6RcY/tripleo-heat-templates
```

```
Invoking workflow (tripleo.derive_params.v1.derive_parameters) specified in plan-environment file
```

```
Started Mistral Workflow tripleo.derive_params.v1.derive_parameters.
```

```
Execution ID: 2d4572bf-4c5b-41f8-8981-c84a363dd95b
```

```
Workflow execution is completed. result:
```

```
ComputeOvsDpdkParameters:
```

```
  IsolCpusList:
```

```
1,2,3,4,5,6,7,9,10,17,18,19,20,21,22,23,11,12,13,14,15,25,26,27,28,29,30,31
```

```
  KernelArgs: default_hugepagesz=1GB hugepagesz=1G hugepages=32 iommu=pt intel_iommu=on
```

```
isolcpus=1,2,3,4,5,6,7,9,10,17,18,19,20,21,22,23,11,12,13,14,15,25,26,27,28,29,30,31
```

```
  NovaReservedHostMemory: 4096
```

```
  NovaVcpuPinSet:
```

```
2,3,4,5,6,7,18,19,20,21,22,23,10,11,12,13,14,15,26,27,28,29,30,31
```

```
  OvsDpdkCoreList: 0,16,8,24
```

```
  OvsDpdkMemoryChannels: 4
```

```
  OvsDpdkSocketMemory: 1024,1024
```

```
  OvsPmdCoreList: 1,17,9,25
```



NOTE

The `OvsDpdkMemoryChannels` parameter cannot be derived from introspection details. In most cases, this value should be 4.

Deploy the Overcloud with the Derived Parameters

To deploy the overcloud with these derived parameters:

1. Copy the derived parameters from the `plan-environment.yaml` to the `network-environment.yaml` file.

```
# DPDK compute node.
ComputeOvsDpdkParameters:
  KernelArgs: default_hugepagesz=1GB hugepagesz=1G hugepages=32
iommu=pt intel_iommu=on
  TunedProfileName: "cpu-partitioning"
  IsolCpusList:
"1,2,3,4,5,6,7,9,10,17,18,19,20,21,22,23,11,12,13,14,15,25,26,27,28,29,30,31"
  NovaVcpuPinSet:
[ '2,3,4,5,6,7,18,19,20,21,22,23,10,11,12,13,14,15,26,27,28,29,30,31'
]
  NovaReservedHostMemory: 4096
  OvsDpdkSocketMemory: "1024,1024"
  OvsDpdkMemoryChannels: "4"
  OvsDpdkCoreList: "0,16,8,24"
  OvsPmdCoreList: "1,17,9,25"
```

**NOTE**

You must assign at least one CPU (with sibling thread) on each NUMA node with or without DPDK NICs present for DPDK PMD to avoid failures in creating guest instances.

**NOTE**

These parameters apply to the specific role (ComputeOvsDpdk). You can apply these parameters globally, but any global parameters are overwritten by role-specific parameters.

2. Deploy the overcloud.

```
#!/bin/bash

openstack overcloud deploy \
--templates \
-r /home/stack/ospd-12-sriov-dpdk-heterogeneous-cluster/roles_data.yaml \
-e /usr/share/openstack-tripleo-heat-templates/environments/network-
isolation.yaml \
-e /usr/share/openstack-tripleo-heat-templates/environments/host-config-
and-reboot.yaml \
-e /usr/share/openstack-tripleo-heat-templates/environments/neutron-
sriov.yaml \
-e /usr/share/openstack-tripleo-heat-templates/environments/neutron-ovs-
dpdk.yaml \
-e /home/stack/ospd-12-sriov-dpdk-heterogeneous-cluster/network-
environment.yaml
```

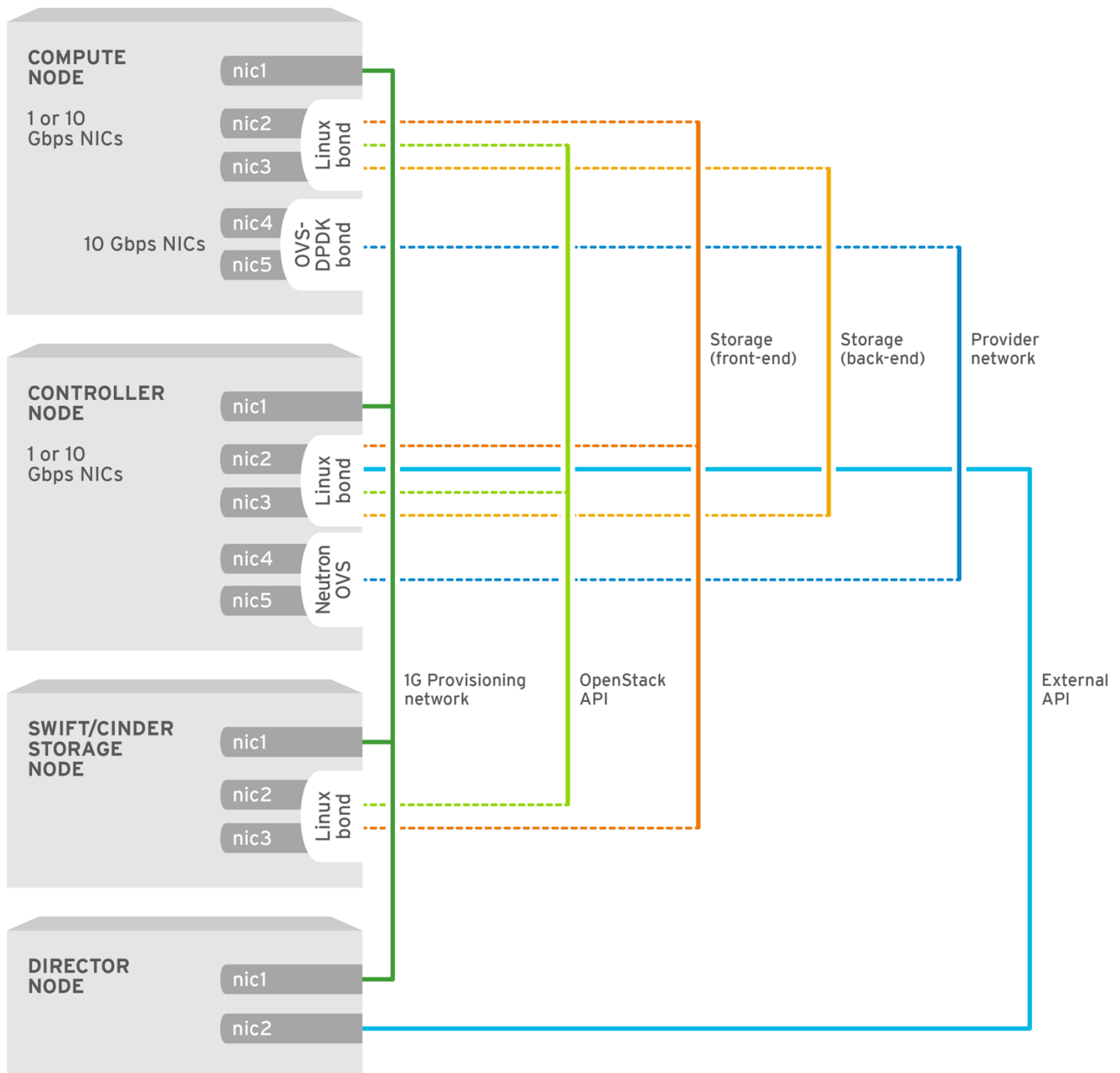
NOTE

In a cluster with Compute, ComputeOvsDpdk and ComputeSriov, the existing derive parameters workflow applies the formula only for the ComputeOvsDpdk role and the other roles are not affected.

8.2. OVS-DPDK TOPOLOGY

With Red Hat OpenStack Platform, you can create custom deployment roles, using the composable roles feature, adding or removing services from each role. For more information on Composable Roles, see [Composable Roles and Services](#).

This image shows a sample OVS-DPDK topology with two bonded ports for the control plane and data plane:



OPENSTACK_450694_0617

Configuring OVS-DPDK comprises the following tasks:

- If you use composable roles, copy and modify the `roles_data.yaml` file to add the custom role for OVS-DPDK.
- Update the appropriate `network-environment.yaml` file to include parameters for kernel arguments and DPDK arguments.
- Update the `compute.yaml` file to include the bridge for DPDK interface parameters.
- Update the `controller.yaml` file to include the same bridge details for DPDK interface parameters.
- Run the `overcloud_deploy.sh` script to deploy the overcloud with the DPDK parameters.

**NOTE**

This guide provides examples for CPU assignments, memory allocation, and NIC configurations that may vary from your topology and use case. See the [Network Functions Virtualization Product Guide](#) and [Chapter 2, Hardware requirements](#) to understand the hardware and configuration options.

Before you begin the procedure, ensure that you have the following:

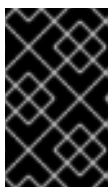
- Red Hat OpenStack Platform 12 with Red Hat Enterprise Linux 7.4
- OVS 2.7
- DPDK 16.11
- Tested NIC. For a list of tested NICs for NFV, see [Section 2.1, “Tested NICs”](#).

**NOTE**

Red Hat OpenStack Platform 12 operates in OVS client mode for OVS-DPDK deployments.

8.3. CONFIGURING TWO-PORT OVS-DPDK DATA PLANE BONDING WITH VLAN TUNNELLING

This section describes how to configure OVS-DPDK with two data plane ports in an OVS-DPDK bond.

**IMPORTANT**

You must determine the best values for the OVS-DPDK parameters that you set in the `network-environment.yaml` file to optimize your OpenStack network for OVS-DPDK. See [Section 8.1, “Deriving DPDK parameters with workflows”](#) for details.

8.3.1. Generating the ComputeOvsDpdk composable role

Generate `roles_data.yaml` for the `ComputeOvsDpdk` role.

```
# openstack overcloud roles generate --roles-path templates/openstack-
tripleo-heat-templates/roles -o roles_data.yaml Controller ComputeOvsDpdk
```

8.3.2. Configuring tuned for CPU affinity

This example uses the sample [post-install.yaml](#) file.

1. Set the `tuned` configuration to enable CPU affinity.

```
resources:

  ExtraDeployments:
    type: OS::Heat::StructuredDeployments
    properties:
      servers: {get_param: servers}
      config: {get_resource: ExtraConfig}
```



```

        actions: ['CREATE', 'UPDATE']

ExtraConfig:
  type: OS::Heat::SoftwareConfig
  properties:
    group: script
    config: |
      #!/bin/bash
      set -x
      function tuned_service_dependency() {
        tuned_service=/usr/lib/systemd/system/tuned.service
        grep -q "network.target" $tuned_service
        if [ "$?" -eq 0 ]; then
          sed -i '/After=.*s/network.target//g'
$tuned_service
        fi
        grep -q "Before=.*network.target" $tuned_service
        if [ ! "$?" -eq 0 ]; then
          grep -q "Before=.*" $tuned_service
          if [ "$?" -eq 0 ]; then
            sed -i 's/^\(Before=.*\)\/\1 network.target
openvswitch.service/g' $tuned_service
          else
            sed -i '/After/i Before=network.target
openvswitch.service' $tuned_service
          fi
        fi
      }

      if hiera -c /etc/puppet/hiera.yaml service_names | grep -q
neutron_ovs_dpdk_agent; then
        tuned_service_dependency
      fi

```

8.3.3. Configuring the OVS-DPDK parameters

This example uses the sample [network-environment.yaml](#) file.



IMPORTANT

You must determine the best values for the OVS-DPDK parameters that you set in the `network-environment.yaml` file to optimize your OpenStack network for OVS-DPDK. See [Section 8.1, “Deriving DPDK parameters with workflows”](#) for details.

1. Add the custom resources for OVS-DPDK under `resource_registry`:

```

resource_registry:
  # Specify the relative/absolute path to the config files you
  # want to use for override the default.
  OS::TripleO::ComputeOvsDpdk::Net::SoftwareConfig: nic-
  configs/computeovsdpdk.yaml
  OS::TripleO::Controller::Net::SoftwareConfig: nic-
  configs/controller.yaml
  OS::TripleO::NodeExtraConfigPost: post-install.yaml

```

2. Under `parameter_defaults`, disable the tunnel type (set the value to `''`), and set the network type to `vlan`:

```
NeutronTunnelTypes: ''
NeutronNetworkType: 'vlan'
```

3. Under `parameter_defaults`, map the physical network to the virtual bridge:

```
NeutronBridgeMappings: 'tenant:br-link0'
```

4. Under `parameter_defaults`, set the OpenStack Networking ML2 and Open vSwitch VLAN mapping range:

```
NeutronNetworkVLANRanges: 'tenant:22:22,tenant:25:25'
```

5. Under `parameter_defaults`, set the role-specific parameters for the `ComputeOvsDpdk` role:

```
#####
# OVS DPDK configuration #
# #####
ComputeOvsDpdkParameters:
  KernelArgs: default_hugepagesz=1GB hugepagesz=1G hugepages=32
iommu=pt intel_iommu=on
  TunedProfileName: "cpu-partitioning"
  IsolCpusList:
"1,2,3,4,5,6,7,9,10,17,18,19,20,21,22,23,11,12,13,14,15,25,26,27,28,
29,30,31"
  NovaVcpuPinSet:
['2,3,4,5,6,7,18,19,20,21,22,23,10,11,12,13,14,15,26,27,28,29,30,31'
]
  NovaReservedHostMemory: 4096
  OvsDpdkSocketMemory: "1024,1024"
  OvsDpdkMemoryChannels: "4"
  OvsDpdkCoreList: "0,16,8,24"
  OvsPmdCoreList: "1,17,9,25"
```



NOTE

You must assign at least one CPU (with sibling thread) on each NUMA node with or without DPDK NICs present for DPDK PMD to avoid failures in creating guest instances.

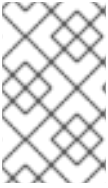


NOTE

These huge pages are consumed by the virtual machines, and also by OVS-DPDK using the `OvsDpdkSocketMemory` parameter as shown in this procedure. The number of huge pages available for the virtual machines is the `boot` parameter minus the `OvsDpdkSocketMemory`.

You must also add `hw:mem_page_size=1GB` to the flavor you associate with the DPDK instance.

+

**NOTE**

OvsDPDKCoreList and **OvsDpdkMemoryChannels** are the required settings for this procedure. Attempting to deploy DPDK without appropriate values causes the deployment to fail or lead to unstable deployments.

1. Under **parameter_defaults**, set the vhost-user socket directory for OVS:

```
NeutronVhostuserSocketDir: "/var/lib/vhost_sockets"
```

2. Set the DHCP Metadata parameters:

```
# DHCP provide metadata route to VM.
NeutronEnableIsolatedMetadata: true
# DHCP always provides metadata route to VM.
NeutronEnableForceMetadata: true
```

8.3.4. Configuring the Controller node

This example uses the sample [controller.yaml](#) file.

1. Create the control plane Linux bond for an isolated network.

```
- type: linux_bond
  name: bond_api
  bonding_options: "mode=active-backup"
  use_dhcp: false
  dns_servers:
    get_param: DnsServers
  addresses:
  - ip_netmask:
    list_join:
      - /
      - - get_param: ControlPlaneIp
        - get_param: ControlPlaneSubnetCidr
  routes:
  - ip_netmask: 169.254.169.254/32
    next_hop:
      get_param: EC2MetadataIp
  members:
  - type: interface
    name: nic2
    primary: true
  - type: interface
    name: nic3
```

2. Assign VLANs to this Linux bond.

```
- type: vlan
  vlan_id:
    get_param: InternalApiNetworkVlanID
  device: bond_api
```

```

addresses:
- ip_netmask:
    get_param: InternalApiIpSubnet

- type: vlan
  vlan_id:
    get_param: TenantNetworkVlanID
  device: bond_api
  addresses:
  - ip_netmask:
      get_param: TenantIpSubnet

- type: vlan
  vlan_id:
    get_param: StorageNetworkVlanID
  device: bond_api
  addresses:
  - ip_netmask:
      get_param: StorageIpSubnet

- type: vlan
  vlan_id:
    get_param: StorageMgmtNetworkVlanID
  device: bond_api
  addresses:
  - ip_netmask:
      get_param: StorageMgmtIpSubnet

- type: vlan
  vlan_id:
    get_param: ExternalNetworkVlanID
  device: bond_api
  addresses:
  - ip_netmask:
      get_param: ExternalIpSubnet
  routes:
  - default: true
    next_hop:
      get_param: ExternalInterfaceDefaultRoute

```

3. Create the OVS bridge for access to the floating IPs into cloud networks.

```

- type: ovs_bridge
  name: br-link0
  use_dhcp: false
  members:
  - type: interface
    name: nic4
    mtu: 9000

```

8.3.5. Configuring the Compute node for DPDK interfaces

This example uses the sample [compute.yaml](#) file.

1. Create the control plane Linux bond for an isolated network.

```

- type: linux_bond
  name: bond_api
  bonding_options: "mode=active-backup"
  use_dhcp: false
  dns_servers:
    get_param: DnsServers
  members:
    - type: interface
      name: nic3
      primary: true
    - type: interface
      name: nic4

```

2. Assign VLANs to this Linux bond.

```

- type: vlan
  vlan_id:
    get_param: InternalApiNetworkVlanID
  device: bond_api
  addresses:
    - ip_netmask:
        get_param: InternalApiIpSubnet

- type: vlan
  vlan_id:
    get_param: TenantNetworkVlanID
  device: bond_api
  addresses:
    - ip_netmask:
        get_param: TenantIpSubnet

- type: vlan
  vlan_id:
    get_param: StorageNetworkVlanID
  device: bond_api
  addresses:
    - ip_netmask:
        get_param: StorageIpSubnet

```

3. Set a bridge with two DPDK ports in an OVS-DPDK bond to link to the controller.

```

- type: ovs_user_bridge
  name: br-link0
  use_dhcp: false
  members:
    - type: ovs_dpdk_bond
      name: dpdkbond0
      mtu: 9000
      rx_queue: 2
      members:
        - type: ovs_dpdk_port
          name: dpdk0
          members:
            - type: interface
              name: nic5

```

```
- type: ovs_dpdk_port
  name: dpdk1
  members:
    - type: interface
      name: nic6
```

**NOTE**

To include multiple DPDK devices, repeat the **type** code section for each DPDK device you want to add.

**NOTE**

When using OVS-DPDK, all bridges on the same Compute node should be of type **ovs_user_bridge**. The director may accept the configuration, but Red Hat OpenStack Platform does not support mixing **ovs_bridge** and **ovs_user_bridge** on the same node.

8.3.6. Deploying the overcloud

Run the [overcloud_deploy.sh](#) script to deploy the overcloud.

```
#!/bin/bash

openstack overcloud deploy \
  --templates \
  -r /home/stack/ospd-12-vlan-dpdk-two-ports-ctlplane-dataplane-
bonding/roles_data.yaml \
  -e /usr/share/openstack-tripleo-heat-templates/environments/network-
isolation.yaml \
  -e /usr/share/openstack-tripleo-heat-templates/environments/host-config-
and-reboot.yaml \
  -e /usr/share/openstack-tripleo-heat-templates/environments/neutron-ovs-
dpdk.yaml \
  -e /home/stack/ospd-12-vlan-dpdk-two-ports-ctlplane-dataplane-
bonding/docker-images.yaml \
  -e /home/stack/ospd-12-vlan-dpdk-two-ports-ctlplane-dataplane-
bonding/network-environment.yaml
```

8.4. CONFIGURING OVS-DPDK WITH VXLAN TUNNELLING

This section describes how to configure OVS-DPDK with VXLAN tunnelling.

**IMPORTANT**

You must determine the best values for the OVS-DPDK parameters that you set in the **network-environment.yaml** file to optimize your OpenStack network for OVS-DPDK. See [Section 8.1, “Deriving DPDK parameters with workflows”](#) for details.

8.4.1. Generating the ComputeOvsDpdk composable role

Generate **roles_data.yaml** for the **ComputeOvsDpdk** role.

-

```
# openstack overcloud roles generate --roles-path templates/openstack-
tripleo-heat-templates/roles -o roles_data.yaml Controller ComputeOvsDpdk
```

8.4.2. Configuring tuned for CPU affinity

This example uses the sample [post-install.yaml](#) file.

1. Set the tuned configuration to enable CPU affinity.

```
resources:

  ExtraDeployments:
    type: OS::Heat::StructuredDeployments
    properties:
      servers: {get_param: servers}
      config: {get_resource: ExtraConfig}
      actions: ['CREATE', 'UPDATE']

  ExtraConfig:
    type: OS::Heat::SoftwareConfig
    properties:
      group: script
      config: |
        #!/bin/bash
        set -x
        function tuned_service_dependency() {
          tuned_service=/usr/lib/systemd/system/tuned.service
          grep -q "network.target" $tuned_service
          if [ "$?" -eq 0 ]; then
            sed -i '/After=.*s/network.target//g'
            $tuned_service
          fi
          grep -q "Before=.*network.target" $tuned_service
          if [ ! "$?" -eq 0 ]; then
            grep -q "Before=.*" $tuned_service
            if [ "$?" -eq 0 ]; then
              sed -i 's/^\(Before=.*\)\/\1 network.target
openvswitch.service/g' $tuned_service
            else
              sed -i '/After/i Before=network.target
openvswitch.service' $tuned_service
            fi
          fi
        }

        if hiera -c /etc/puppet/hiera.yaml service_names | grep -q
neutron_ovs_dpdk_agent; then
          tuned_service_dependency
        fi
```

8.4.3. Configuring OVS-DPDK parameters

This example uses the sample [network-environment.yaml](#) file.

**IMPORTANT**

You must determine the best values for the OVS-DPDK parameters that you set in the `network-environment.yaml` file to optimize your OpenStack network for OVS-DPDK. See [Section 8.1, “Deriving DPDK parameters with workflows”](#) for details.

1. Add the custom resources for OVS-DPDK under `resource_registry`:

```
resource_registry:
  # Specify the relative/absolute path to the config files you
  # want to use for override the default.
  OS::TripleO::ComputeOvsDpdk::Net::SoftwareConfig: nic-
  configs/compute-ovs-dpdk.yaml
  OS::TripleO::Controller::Net::SoftwareConfig: nic-
  configs/controller.yaml
  OS::TripleO::NodeExtraConfigPost: post-install.yaml
```

2. Under `parameter_defaults`, set the tunnel type and the tenant type to `vxlan`:

```
NeutronTunnelTypes: 'vxlan'
NeutronNetworkType: 'vxlan'
```

3. Under `parameter_defaults`, set the role-specific parameters for the `ComputeOvsDpdk` role:

```
# DPDK compute node.
ComputeOvsDpdkParameters:
  KernelArgs: default_hugepagesz=1GB hugepagesz=1G hugepages=32
  iommu=pt intel_iommu=on
  TunedProfileName: "cpu-partitioning"
  IsolCpusList:
    "1,2,3,4,5,6,7,9,10,17,18,19,20,21,22,23,11,12,13,14,15,25,26,27,28,
    29,30,31"
  NovaVcpuPinSet:
    ['2,3,4,5,6,7,18,19,20,21,22,23,10,11,12,13,14,15,26,27,28,29,30,31']
  NovaReservedHostMemory: 4096
  OvsDpdkSocketMemory: "1024,1024"
  OvsDpdkMemoryChannels: "4"
  OvsDpdkCoreList: "0,16,8,24"
  OvsPmdCoreList: "1,17,9,25"
```

**NOTE**

You must assign at least one CPU (with sibling thread) on each NUMA node with or without DPDK NICs present for DPDK PMD to avoid failures in creating guest instances.

**NOTE**

These huge pages are consumed by the virtual machines, and also by OVS-DPDK using the `OvsDpdkSocketMemory` parameter as shown in this procedure. The number of huge pages available for the virtual machines is the `boot` parameter minus the `OvsDpdkSocketMemory`.

You must also add `hw:mem_page_size=1GB` to the flavor you associate with the DPDK instance.

**NOTE**

`OvsDPDKCoreList` and `OvsDpdkMemoryChannels` are the **required** settings for this procedure. Attempting to deploy DPDK without appropriate values causes the deployment to fail or lead to unstable deployments.

4. Under `parameter_defaults`, set the vhost-user socket directory for OVS:

```
NeutronVhostuserSocketDir: "/var/lib/vhost_sockets"
```

5. Set the DHCP Metadata parameters:

```
# DHCP provide metadata route to VM.
NeutronEnableIsolatedMetadata: true
# DHCP always provides metadata route to VM.
NeutronEnableForceMetadata: true
```

8.4.4. Configuring the Controller node

This example uses the sample `controller.yaml` file.

1. Create the control plane Linux bond for an isolated network.

```
- type: linux_bond
  name: bond_api
  bonding_options: "mode=active-backup"
  use_dhcp: false
  dns_servers:
    get_param: DnsServers
  addresses:
  - ip_netmask:
    list_join:
      - /
      - - get_param: ControlPlaneIp
        - get_param: ControlPlaneSubnetCidr
  routes:
  - ip_netmask: 169.254.169.254/32
    next_hop:
      get_param: EC2MetadataIp
  members:
  - type: interface
    name: nic2
```

```

    primary: true
  - type: interface
    name: nic3

```

2. Assign VLANs to this Linux bond.

```

- type: vlan
  vlan_id:
    get_param: InternalApiNetworkVlanID
  device: bond_api
  addresses:
  - ip_netmask:
    get_param: InternalApiIpSubnet

- type: vlan
  vlan_id:
    get_param: StorageNetworkVlanID
  device: bond_api
  addresses:
  - ip_netmask:
    get_param: StorageIpSubnet

- type: vlan
  vlan_id:
    get_param: StorageMgmtNetworkVlanID
  device: bond_api
  addresses:
  - ip_netmask:
    get_param: StorageMgmtIpSubnet

- type: vlan
  vlan_id:
    get_param: ExternalNetworkVlanID
  device: bond_api
  addresses:
  - ip_netmask:
    get_param: ExternalIpSubnet
  routes:
  - default: true
    next_hop:
      get_param: ExternalInterfaceDefaultRoute

```

3. Create the OVS bridge for access to the floating IPs into cloud networks.

```

- type: ovs_bridge
  name: br-link0
  use_dhcp: false
  members:
  - type: interface
    name: nic4
    mtu: 9000
  - type: vlan
    vlan_id:
      get_param: TenantNetworkVlanID
    addresses:
  - ip_netmask:

```

```
get_param: TenantIpSubnet
```

8.4.5. Configuring the Compute node for DPDK interfaces

This example uses the `compute.yaml` file.

Create the `compute-ovs-dpdk.yaml` file from the default `compute.yaml` file and make the following changes:

1. Create the control plane Linux bond for an isolated network.

```
- type: linux_bond
  name: bond_api
  bonding_options: "mode=active-backup"
  use_dhcp: false
  dns_servers:
    get_param: DnsServers
  members:
    - type: interface
      name: nic3
      primary: true
    - type: interface
      name: nic4
```

2. Assign VLANs to this Linux bond.

```
- type: vlan
  vlan_id:
    get_param: InternalApiNetworkVlanID
  device: bond_api
  addresses:
    - ip_netmask:
        get_param: InternalApiIpSubnet

- type: vlan
  vlan_id:
    get_param: StorageNetworkVlanID
  device: bond_api
  addresses:
    - ip_netmask:
        get_param: StorageIpSubnet
```

3. Set a bridge with a DPDK port to link to the controller.

```
- type: ovs_user_bridge
  name: br-link0
  use_dhcp: false
  ovs_extra:
    - str_replace:
        template: set port br-link0 tag=_VLAN_TAG_
        params:
          _VLAN_TAG_:
            get_param: TenantNetworkVlanID
  addresses:
    - ip_netmask:
```

```

        get_param: TenantIpSubnet
members:
- type: ovs_dpdk_bond
  name: dpdkbond0
  mtu: 9000
  rx_queue: 2
  members:
    - type: ovs_dpdk_port
      name: dpdk0
      members:
        - type: interface
          name: nic5
    - type: ovs_dpdk_port
      name: dpdk1
      members:
        - type: interface
          name: nic6

```

**NOTE**

To include multiple DPDK devices, repeat the **type** code section for each DPDK device you want to add.

**NOTE**

When using OVS-DPDK, all bridges on the same Compute node should be of type **ovs_user_bridge**. The director may accept the configuration, but Red Hat OpenStack Platform does not support mixing **ovs_bridge** and **ovs_user_bridge** on the same node.

8.4.6. Deploying the overcloud

Run the [overcloud_deploy.sh](#) script to deploy the overcloud.

```

#!/bin/bash

openstack overcloud deploy \
--templates \
-r /home/stack/ospd-12-vxlan-dpdk-single-port-ctlplane-
bonding/roles_data.yaml \
-e /usr/share/openstack-tripleo-heat-templates/environments/network-
isolation.yaml \
-e /usr/share/openstack-tripleo-heat-templates/environments/host-config-
and-reboot.yaml \
-e /usr/share/openstack-tripleo-heat-templates/environments/neutron-ovs-
dpdk.yaml \
-e /home/stack/ospd-12-vxlan-dpdk-single-port-ctlplane-bonding/docker-
images.yaml \
-e /home/stack/ospd-12-vxlan-dpdk-single-port-ctlplane-bonding/network-
environment.yaml

```

8.5. SETTING THE MTU VALUE FOR OVS-DPDK INTERFACES

Red Hat OpenStack Platform supports jumbo frames for OVS-DPDK. To set the MTU value for jumbo frames you must:

- Set the global MTU value for networking in the `network-environment.yaml` file.
- Set the physical DPDK port MTU value in the `compute.yaml` file. This value is also used by the vhost user interface.
- Set the MTU value within any guest instances on the Compute node to ensure that you have a comparable MTU value from end to end in your configuration.



NOTE

VXLAN packets include an extra 50 bytes in the header. Calculate your MTU requirements based on these additional header bytes. For example, an MTU value of 9000 means the VXLAN tunnel MTU value is 8950 to account for these extra bytes.



NOTE

You do not need any special configuration for the physical NIC since the NIC is controlled by the DPDK PMD and has the same MTU value set by the `compute.yaml` file. You cannot set an MTU value larger than the maximum value supported by the physical NIC.

To set the MTU value for OVS-DPDK interfaces:

1. Set the `NeutronGlobalPhysnetMtu` parameter in the `network-environment.yaml` file.

```
parameter_defaults:
  # MTU global configuration
  NeutronGlobalPhysnetMtu: 9000
```



NOTE

Ensure that the `NeutronDpdkSocketMemory` value in the `network-environment.yaml` file is large enough to support jumbo frames. See [Section 7.4.2, “Memory parameters”](#) for details.

2. Set the MTU value on the bridge to the Compute node in the `controller.yaml` file.

```
-
  type: ovs_bridge
  name: br-link0
  use_dhcp: false
  members:
    -
      type: interface
      name: nic3
      mtu: 9000
```

3. Set the MTU values for an OVS-DPDK bond in the `compute.yaml` file:

```
- type: ovs_user_bridge
```

```

name: br-link0
use_dhcp: false
members:
  - type: ovs_dpdk_bond
    name: dpdkbond0
    mtu: 9000
    rx_queue: 2
    members:
      - type: ovs_dpdk_port
        name: dpdk0
        mtu: 9000
        members:
          - type: interface
            name: nic4
      - type: ovs_dpdk_port
        name: dpdk1
        mtu: 9000
        members:
          - type: interface
            name: nic5

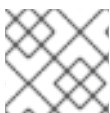
```

8.6. CONFIGURING SECURITY GROUPS

You can configure security groups to use the OVS firewall driver in Red Hat OpenStack Platform director. The `NeutronOVSFirewallDriver` parameter allows you to control which firewall driver is used:

- **iptables_hybrid** - Configures Networking to use the iptables/hybrid based implementation.
- **openvswitch** - Configures Networking to use the OVS firewall flow-based driver.

The **openvswitch** OVS firewall driver includes higher performance and reduces the number of interfaces and bridges used to connect guests to the project network.



NOTE

The **iptables_hybrid** option is not compatible with OVS-DPDK.

Configure the `NeutronOVSFirewallDriver` parameter in the `network-environment.yaml` file:

```

# Configure the classname of the firewall driver to use for implementing
# security groups.
NeutronOVSFirewallDriver: openvswitch

```

See [Basic security group operations](#) for details on enabling and disabling security groups.

8.7. SETTING MULTIQUEUE FOR OVS-DPDK INTERFACES

To set same number of queues for interfaces in OVS-DPDK on the Compute node, modify the `compute.yaml` file as follows:

```

- type: ovs_user_bridge
  name: br-link0

```

```

use_dhcp: false
members:
  - type: ovs_dpdk_bond
    name: dpdkbond0
    mtu: 9000
    rx_queue: 2
    members:
      - type: ovs_dpdk_port
        name: dpdk0
        mtu: 9000
        members:
          - type: interface
            name: nic4
      - type: ovs_dpdk_port
        name: dpdk1
        mtu: 9000
        members:
          - type: interface
            name: nic5

```

8.8. KNOWN LIMITATIONS

There are certain limitations when configuring OVS-DPDK with Red Hat OpenStack Platform for the NFV use case:

- Use Linux bonds for control plane networks. Ensure both PCI devices used in the bond are on the same NUMA node for optimum performance. Neutron Linux bridge configuration is not supported by Red Hat.
- Huge pages are required for every instance running on the hosts with OVS-DPDK. If huge pages are not present in the guest, the interface appears but does not function.
- There is a performance degradation of services that use tap devices, because these devices do not support DPDK. For example, services such as DVR, FWaaS, and LBaaS use tap devices.
 - With OVS-DPDK, you can enable DVR with `netdev datapath`, but this has poor performance and is not suitable for a production environment. DVR uses kernel namespace and tap devices to perform the routing.
 - To ensure the DVR routing performs well with OVS-DPDK, you need to use a controller such as ODL which implements routing as OpenFlow rules. With OVS-DPDK, OpenFlow routing removes the bottleneck introduced by the Linux kernel interfaces so that the full performance of datapath is maintained.
- When using OVS-DPDK, all bridges on the same Compute node should be of type `ovs_user_bridge`. The director may accept the configuration, but Red Hat OpenStack Platform does not support mixing `ovs_bridge` and `ovs_user_bridge` on the same node.

8.9. CREATING A FLAVOR AND DEPLOYING AN INSTANCE FOR OVS-DPDK

After you have completed configuring OVS-DPDK for your Red Hat OpenStack Platform deployment with NFV, you can create a flavor and deploy an instance with the following steps:

1. Create an aggregate group and add a host to it for OVS-DPDK.

```
# openstack aggregate create --zone=dppdk dpdk
# openstack aggregate add host dpdk compute-ovs-dpdk-0.localdomain
```

**NOTE**

You should use host aggregates to separate CPU pinned instances from unpinned instances. Instances that do not use CPU pinning do not respect the resourcing requirements of instances that use CPU pinning.

2. Create a flavor.

```
# openstack flavor create m1.medium_huge_4cpu --ram 4096 --disk 150
--vcpus 4
```

Here, **m1.medium_huge_4cpu** is the flavor name, **4096** is the memory size in MB, **150** is the disk size in GB (default 0G), and **4** is the number of vCPUs.

3. Set additional flavor properties.

```
# openstack flavor set --property hw:cpu_policy=dedicated --property
hw:mem_page_size=1GB m1.medium_huge_4cpu --property
hw:emulator_threads_policy=isolate
```

Here, **m1.medium_huge_4cpu** is the flavor name and the remaining parameters set the other properties for the flavor.

See [Configure Emulator Threads to run on a Dedicated Physical CPU](#) for details on the emulator threads policy for performance improvements.

1. Create the network.

```
# openstack network create net1 --provider-physical-network tenant -
--provider-network-type vlan --provider-segment <VLAN-ID>
```

2. Deploy an instance.

```
# openstack server create --flavor m1.medium_huge_4cpu --
availability-zone dpdk --image rhel_7.3 --nic net-id=net1
```

Where:

- **m1.medium_huge_4cpu** is the flavor name or ID.
- **dpdk** is the availability zone for the server.
- **rhel_7.3** is the image (name or ID) used to create an instance.
- **net1** is the NIC on the server.

You have now deployed an instance for the OVS-DPDK with NFV use case.

For using **multi-queue** with OVS-DPDK, there are a couple of additional steps that you need to include in the above procedure. Before you create a flavor, perform the following steps:

1. Set the image properties.

```
# openstack image set --property hw_vif_multiqueue_enabled=true
<image-id>
```

Here, `hw_vif_multiqueue_enabled=true` is a property on this image to enable multiqueue, `<image-id>` is the name or ID of the image to modify.

2. Set additional flavor properties.

```
# openstack flavor set m1.vm_mq set hw:vif_multiqueue_enabled=true
```

Here, `m1.vm_mq` is the flavor ID or name, and the remaining options enable multiqueue for the flavor.

8.10. TROUBLESHOOTING THE CONFIGURATION

This section describes the steps to troubleshoot the DPDK-OVS configuration.

1. Review the bridge configuration and confirm that the bridge was created with the `datapath_type=netdev`.

```
# ovs-vsctl list bridge br0
_uuid                : bdce0825-e263-4d15-b256-f01222df96f3
auto_attach          : []
controller            : []
datapath_id           : "00002608cebd154d"
datapath_type         : netdev
datapath_version      : "<built-in>"
external_ids          : {}
fail_mode             : []
flood_vlans           : []
flow_tables           : {}
ipfix                 : []
mcast_snooping_enable: false
mirrors              : []
name                  : "br0"
netflow               : []
other_config          : {}
ports                 : [52725b91-de7f-41e7-bb49-3b7e50354138]
protocols             : []
rstp_enable           : false
rstp_status           : {}
sflow                : []
status                : {}
stp_enable            : false
```

2. Review the OVS service by confirming that the `neutron-ovs-agent` is configured to start automatically.

```
# systemctl status neutron-openvswitch-agent.service
neutron-openvswitch-agent.service - OpenStack Neutron Open vSwitch
Agent
Loaded: loaded (/usr/lib/systemd/system/neutron-openvswitch-
```

```
agent.service; enabled; vendor preset: disabled)
Active: active (running) since Mon 2015-11-23 14:49:31 AEST; 25min
ago
```

If the service is having trouble starting, you can view any related messages.

```
# journalctl -t neutron-openvswitch-agent.service
```

3. Confirm that the PMD CPU mask of the **ovs-dpdk** are pinned to the CPUs. In case of HT, use sibling CPUs.

For example, take **CPU4**:

```
# cat /sys/devices/system/cpu/cpu4/topology/thread_siblings_list
4,20
```

So, using CPU 4 and 20:

```
# ovs-vsctl set Open_vSwitch . other_config:pmd-cpu-mask=0x100010
```

Display their status:

```
# tuna -t ovs-vswitchd -CP
thread  ctxt_switches pid SCHED_ rtpri affinity voluntary
nonvoluntary      cmd
3161 OTHER  0      6 765023      614 ovs-vswitchd
3219  OTHER  0      6      1      0 handler24
3220  OTHER  0      6      1      0 handler21
3221  OTHER  0      6      1      0 handler22
3222  OTHER  0      6      1      0 handler23
3223  OTHER  0      6      1      0 handler25
3224  OTHER  0      6      1      0 handler26
3225  OTHER  0      6      1      0 handler27
3226  OTHER  0      6      1      0 handler28
3227  OTHER  0      6      2      0 handler31
3228  OTHER  0      6      2      4 handler30
3229  OTHER  0      6      2      5 handler32
3230  OTHER  0      6 953538      431 revalidator29
3231  OTHER  0      6 1424258      976 revalidator33
3232  OTHER  0      6 1424693      836 revalidator34
3233  OTHER  0      6 951678      503 revalidator36
3234  OTHER  0      6 1425128      498 revalidator35
*3235  OTHER  0      4 151123      51 pmd37*
*3236  OTHER  0     20 298967      48 pmd38*
3164  OTHER  0      6 47575      0 dpdk_watchdog3
3165  OTHER  0      6 237634      0 vhost_thread1
3166  OTHER  0      6 3665      0 urcu2
```

CHAPTER 9. CONFIGURING A HETEROGENEOUS COMPUTE CLUSTER

This section describes how to deploy an SR-IOV Compute node and an OVS-DPDK Compute node in the same environment. This deployment uses custom roles for OVS-DPDK and SR-IOV with role-specific parameters defined in the `network-environment.yaml` file. The process to create and deploy a composable role includes:

- Define the SR-IOV and OVS-DPDK custom roles in a local copy of the `roles_data.yaml` file.
- Set the role-specific parameters for the SR-IOV role and the OVS-DPDK role in the `network_environment.yaml` file.
- Deploy the overcloud with this updated set of roles.

You must install and configure the undercloud before you can deploy a heterogeneous compute cluster in the overcloud. See the [Director Installation and Usage Guide](#) for details.



NOTE

Ensure that you create OpenStack flavors that match these custom roles.



IMPORTANT

You must determine the best values for the OVS-DPDK parameters that you set in the `network-environment.yaml` file to optimize your OpenStack network for OVS-DPDK. See [Section 8.1, “Deriving DPDK parameters with workflows”](#) for details.

9.1. NAMING CONVENTIONS

We recommend that you follow a consistent naming convention when you use custom roles in your OpenStack deployment, especially with a heterogeneous compute cluster. This naming convention can assist you when creating the following files and configurations:

- `instackenv.json` - To differentiate between the compute node that uses SR-IOV interfaces from the compute node that uses DPDK interfaces.

```
"name": "computeovsdpdk-0"
```

- `roles_data.yaml` - To differentiate between compute-based roles that support SR-IOV from compute-based roles that support DPDK.

```
`ComputeOvsDpdk`
```

- `network_environment.yaml` -

- To ensure that you match the custom role to the correct flavor name.

```
`OvercloudComputeOvsDpdkFlavor: computeovsdpdk`
```

- To include the correct custom role name for any role-specific parameters.

```
`ComputeOvsDpdkParameters`
```

- **nic-config** file names - To differentiate NIC yaml files for compute nodes that support SR-IOV from compute nodes that support DPDK interfaces.
- **Flavor creation** - To help you match a flavor and **capabilities:profile** value to the appropriate bare metal node and custom role.

```
# openstack flavor create --id auto --ram 4096 --disk 40 --vcpus 4
computeovsdpdk
# openstack flavor set --property "cpu_arch"="x86_64" --property
"capabilities:boot_option"="local" --property
"capabilities:profile"="computeovsdpdk" computeovsdpdk
```

- **Bare metal node** - To ensure that you match the bare metal node with the appropriate hardware and **capability:profile** value.

```
# openstack baremetal node update computeovsdpdk-0 add
properties/capabilities='profile:computeovsdpdk,boot_option:local'
```



NOTE

The flavor name does not have to match the **capabilities:profile** value for the flavor, but the flavor **capabilities:profile** value must match the bare metal node **properties/capabilities='profile'** value. All three use **computeovsdpdk** in this example.



NOTE

Ensure that all your nodes used for a custom role and profile have the same CPU, RAM, and PCI hardware topology.

In this example, the **ComputeOvsDpdk** and **ComputeSriov** are custom roles for compute nodes to enable DPDK or SR-IOV only on the nodes that have the appropriate NICs. The existing set of default roles provided by Red Hat OpenStack Platform is stored in the **/home/stack/roles_data.yaml** file.

9.2. CREATING THE SR-IOV AND OVS-DPDK CUSTOM ROLES

Red Hat OpenStack provides a set of default roles in the **roles_data.yaml** file. You can create your own **roles_data.yaml** file to support the roles you need.

To create the custom roles to support SR-IOV and OVS-DPDK:

1. Create the **ComputeSriov.yaml** file in a local directory and add the definition of this role:

```
#####
#####
# Role: ComputeSriov
#
#####
#####
- name: ComputeSriov
  description: |
    Compute SR-IOV role
```

```

CountDefault: 1
networks:
  - InternalApi
  - Tenant
  - Storage
HostnameFormatDefault: 'computesriov-%index%'
disable_upgrade_deployment: True
ServicesDefault:
  - OS::Triple0::Services::AuditD
  - OS::Triple0::Services::CACerts
  - OS::Triple0::Services::CephClient
  - OS::Triple0::Services::CephExternal
  - OS::Triple0::Services::CertmongerUser
  - OS::Triple0::Services::Collectd
  - OS::Triple0::Services::ComputeCeilometerAgent
  - OS::Triple0::Services::ComputeNeutronCorePlugin
  - OS::Triple0::Services::ComputeNeutronL3Agent
  - OS::Triple0::Services::ComputeNeutronMetadataAgent
  - OS::Triple0::Services::ComputeNeutronOvsAgent
  - OS::Triple0::Services::Docker
  - OS::Triple0::Services::FluentdClient
  - OS::Triple0::Services::Iscsid
  - OS::Triple0::Services::Kernel
  - OS::Triple0::Services::MySQLClient
  - OS::Triple0::Services::NeutronLinuxbridgeAgent
  - OS::Triple0::Services::NeutronSriovAgent
  - OS::Triple0::Services::NeutronVppAgent
  - OS::Triple0::Services::NovaCompute
  - OS::Triple0::Services::NovaLibvirt
  - OS::Triple0::Services::NovaMigrationTarget
  - OS::Triple0::Services::Ntp
  - OS::Triple0::Services::OpenDaylightOvs
  - OS::Triple0::Services::Securetty
  - OS::Triple0::Services::SensuClient
  - OS::Triple0::Services::Snmp
  - OS::Triple0::Services::Sshd
  - OS::Triple0::Services::Timezone
  - OS::Triple0::Services::TripleoFirewall
  - OS::Triple0::Services::TripleoPackages
  - OS::Triple0::Services::Tuned
  - OS::Triple0::Services::Vpp
  - OS::Triple0::Services::OVNController

```

2. Create the **ComputeOvsDpdk.yaml** file in a local directory and add the definition of this role:

```

#####
#####
# Role: ComputeOvsDpdk
#
#####
#####
- name: ComputeOvsDpdk
  description: |
    Compute OVS DPDK Role
  CountDefault: 1
  networks:

```

```

- InternalApi
- Tenant
- Storage
HostnameFormatDefault: 'computeovsdpsdk-%index%'
disable_upgrade_deployment: True
ServicesDefault:
- OS::TripleO::Services::AuditD
- OS::TripleO::Services::CACerts
- OS::TripleO::Services::CephClient
- OS::TripleO::Services::CephExternal
- OS::TripleO::Services::CertmongerUser
- OS::TripleO::Services::Collectd
- OS::TripleO::Services::ComputeCeilometerAgent
- OS::TripleO::Services::ComputeNeutronCorePlugin
- OS::TripleO::Services::ComputeNeutronL3Agent
- OS::TripleO::Services::ComputeNeutronMetadataAgent
- OS::TripleO::Services::ComputeNeutronOvsDpsdk
- OS::TripleO::Services::Docker
- OS::TripleO::Services::FluentdClient
- OS::TripleO::Services::Iscsid
- OS::TripleO::Services::Kernel
- OS::TripleO::Services::MySQLClient
- OS::TripleO::Services::NovaCompute
- OS::TripleO::Services::NovaLibvirt
- OS::TripleO::Services::Ntp
- OS::TripleO::Services::OpenDaylightOvs
- OS::TripleO::Services::Securetty
- OS::TripleO::Services::SensuClient
- OS::TripleO::Services::Snmp
- OS::TripleO::Services::Sshd
- OS::TripleO::Services::Timezone
- OS::TripleO::Services::TripleoFirewall
- OS::TripleO::Services::TripleoPackages

```

3. Generate `roles_data.yaml` for the **ComputeSriov** and **ComputeOvsDpsdk** roles and any other roles you need for your deployment:

```

# openstack overcloud roles generate --roles-path
templates/openstack-tripleo-heat-templates/roles -o roles_data.yaml
Controller ComputeSriov ComputeOvsDpsdk

```

9.3. CONFIGURING TUNED FOR CPU AFFINITY

This example uses the sample [post-install.yaml](#) file.

1. Set the **tuned** configuration to enable CPU affinity.

```

resources:

  ExtraDeployments:
    type: OS::Heat::StructuredDeployments
    properties:
      servers: {get_param: servers}
      config: {get_resource: ExtraConfig}
      actions: ['CREATE', 'UPDATE']

```

```

ExtraConfig:
  type: OS::Heat::SoftwareConfig
  properties:
    group: script
    config: |
      #!/bin/bash
      set -x
      function tuned_service_dependency() {
        tuned_service=/usr/lib/systemd/system/tuned.service
        grep -q "network.target" $tuned_service
        if [ "$?" -eq 0 ]; then
          sed -i '/After=.*s/network.target//g'
$tuned_service
        fi
        grep -q "Before=.*network.target" $tuned_service
        if [ ! "$?" -eq 0 ]; then
          grep -q "Before=.*" $tuned_service
          if [ "$?" -eq 0 ]; then
            sed -i 's/^\(Before=.*\)\/\1 network.target
openvswitch.service/g' $tuned_service
          else
            sed -i '/After/i Before=network.target
openvswitch.service' $tuned_service
          fi
        fi
      }

      if hiera -c /etc/puppet/hiera.yaml service_names | grep -q
neutron_ovs_dpdk_agent; then
        tuned_service_dependency
      fi

```

9.4. DEFINING THE SR-IOV AND OVS-DPDK ROLE-SPECIFIC PARAMETERS



IMPORTANT

You must determine the best values for the OVS-DPDK parameters that you set in the `network-environment.yaml` file to optimize your OpenStack network for OVS-DPDK. See [Planning your OVS-DPDK Deployment](#) and [Section 8.1, “Deriving DPDK parameters with workflows”](#) for details.

To configure SR-IOV and OVS-DPDK role-specific parameters in the `network-environment.yaml` file:

1. Add the resource mapping for the OVS-DPDK and SR-IOV services to the `network-environment.yaml` file along with the network configuration for these nodes:

```

resource_registry:
  # Specify the relative/absolute path to the config files you
  # want to use for override the default.
  OS::TripleO::ComputeSriov::Net::SoftwareConfig: nic-
configs/compute-sriov.yaml

```

```
OS::TripleO::ComputeOvsDpdk::Net::SoftwareConfig: nic-
configs/compute-ovs-dpdk.yaml
OS::TripleO::Controller::Net::SoftwareConfig: nic-
configs/controller.yaml
OS::TripleO::NodeExtraConfigPost: post-install.yaml
```

2. Specify the flavors for each role:

```
OvercloudControlFlavor: controller
OvercloudComputeOvsDpdkFlavor: computeovsdpdk
OvercloudComputeSriovFlavor: computesriov
```

3. Specify the number of nodes to deploy for each role:

```
#Number of nodes to deploy.
ControllerCount: 1
ComputeOvsDpdkCount: 1
ComputeSriovCount: 1
```

4. Configure the SR-IOV parameters:

- a. Configure the Compute `pci_passthrough_whitelist` parameter, and set `devname` for the SR-IOV interface. The whitelist sets the PCI devices available to instances.

```
NovaPCIPassthrough:
- devname: "ens2f0"
  physical_network: "tenant"
- devname: "ens2f1"
  physical_network: "tenant"
```

- b. Specify the physical network and SR-IOV interface in the format - **PHYSICAL_NETWORK:PHYSICAL_DEVICE:**

All physical networks listed in the `network_vlan_ranges` on the server should have mappings to the appropriate interfaces on each agent.

```
NeutronPhysicalDevMappings: "tenant:ens2f0,tenant:ens2f1"
```

This example uses `tenant` as the `physical_network` name.

- c. Provide the number of Virtual Functions (VFs) to be reserved for each SR-IOV interface:

```
NeutronSriovNumVFs: "ens2f0:5,ens2f1:5"
```

This example reserves 5 VFs for the SR-IOV interface.



NOTE

Red Hat OpenStack Platform supports the number of VFs supported by the NIC vendor. See [Deployment Limits for Red Hat OpenStack Platform](#) for other related details.

5. Configure the role-specific parameters for SR-IOV:


```
# SR-IOV compute node.
ComputeSriovParameters:
  KernelArgs: default_hugepagesz=1GB hugepagesz=1G hugepages=32
iommu=pt intel_iommu=on
  TunedProfileName: "cpu-partitioning"
  IsolCpusList:
"1,2,3,4,5,6,7,9,10,17,18,19,20,21,22,23,11,12,13,14,15,25,26,27,28,
29,30,31"
  NovaVcpuPinSet:
['2,3,4,5,6,7,18,19,20,21,22,23,10,11,12,13,14,15,26,27,28,29,30,31'
]
  NovaReservedHostMemory: 4096
```

6. Configure the role-specific parameters for OVS-DPDK:

```
# DPDK compute node.
ComputeOvsDpdkParameters:
  KernelArgs: default_hugepagesz=1GB hugepagesz=1G hugepages=32
iommu=pt intel_iommu=on
  TunedProfileName: "cpu-partitioning"
  IsolCpusList:
"1,2,3,4,5,6,7,9,10,17,18,19,20,21,22,23,11,12,13,14,15,25,26,27,28,
29,30,31"
  NovaVcpuPinSet:
['2,3,4,5,6,7,18,19,20,21,22,23,10,11,12,13,14,15,26,27,28,29,30,31'
]
  NovaReservedHostMemory: 4096
  OvsDpdkSocketMemory: "1024,1024"
  OvsDpdkMemoryChannels: "4"
  OvsDpdkCoreList: "0,16,8,24"
  OvsPmdCoreList: "1,17,9,25"
```



NOTE

You must assign at least one CPU (with sibling thread) on each NUMA node with or without DPDK NICs present for DPDK PMD to avoid failures in creating guest instances.

7. Set the DHCP Metadata parameters:

```
# DHCP provide metadata route to VM.
NeutronEnableIsolatedMetadata: true
# DHCP always provides metadata route to VM.
NeutronEnableForceMetadata: true
```

8. Configure the remainder of the `network-environment.yaml` file to override the default parameters from the `neutron-ovs-dpdk-agent.yaml` and `neutron-sriov-agent.yaml` files as needed for your OpenStack deployment.

9.5. CONFIGURING SR-IOV AND DPDK COMPUTE NODES

To support an SR-IOV Compute node and an OVS-DPDK Compute node:

1. Create the `compute-sriov.yaml` file to support SR-IOV interfaces.

a. Create the control plane Linux bond for an isolated network:

```
- type: linux_bond
  name: bond_api
  bonding_options: "mode=active-backup"
  use_dhcp: false
  dns_servers:
    get_param: DnsServers
  members:
    - type: interface
      name: nic3
      primary: true
    - type: interface
      name: nic4
```

b. Assign VLANs to this Linux bond:

```
- type: vlan
  vlan_id:
    get_param: InternalApiNetworkVlanID
  device: bond_api
  addresses:
    - ip_netmask:
        get_param: InternalApiIpSubnet

- type: vlan
  vlan_id:
    get_param: TenantNetworkVlanID
  device: bond_api
  addresses:
    - ip_netmask:
        get_param: TenantIpSubnet

- type: vlan
  vlan_id:
    get_param: StorageNetworkVlanID
  device: bond_api
  addresses:
    - ip_netmask:
        get_param: StorageIpSubnet
```

c. Create the SR-IOV interfaces to the Controller:

```
- type: interface
  name: ens2f0
  mtu: 9000
  use_dhcp: false
  defroute: false
  nm_controlled: true
  hotplug: true

- type: interface
  name: ens2f1
```

```

mtu: 9000
use_dhcp: false
defroute: false
nm_controlled: true
hotplug: true

```

2. Create the **compute-ovsdpdk.yaml** file to support DPDK interfaces.

a. Create the control plane Linux bond for an isolated network:

```

- type: linux_bond
  name: bond_api
  bonding_options: "mode=active-backup"
  use_dhcp: false
  dns_servers:
    get_param: DnsServers
  members:
    - type: interface
      name: nic3
      primary: true
    - type: interface
      name: nic4

```

b. Assign VLANs to this Linux bond:

```

- type: vlan
  vlan_id:
    get_param: InternalApiNetworkVlanID
  device: bond_api
  addresses:
    - ip_netmask:
        get_param: InternalApiIpSubnet

- type: vlan
  vlan_id:
    get_param: TenantNetworkVlanID
  device: bond_api
  addresses:
    - ip_netmask:
        get_param: TenantIpSubnet

- type: vlan
  vlan_id:
    get_param: StorageNetworkVlanID
  device: bond_api
  addresses:
    - ip_netmask:
        get_param: StorageIpSubnet

```

c. Set the bridge with an OVS-DPDK bond to link to the Controller:

```

- type: ovs_user_bridge
  name: br-link0
  use_dhcp: false
  members:

```

```

- type: ovs_dpdk_bond
  name: dpdkbond0
  mtu: 9000
  rx_queue: 2
  members:
    - type: ovs_dpdk_port
      name: dpdk0
      mtu: 9000
      members:
        - type: interface
          name: nic5
    - type: ovs_dpdk_port
      name: dpdk1
      mtu: 9000
      members:
        - type: interface
          name: nic6

```

**NOTE**

To include multiple DPDK devices, repeat the **type** code section for each DPDK device you want to add.

**NOTE**

When using OVS-DPDK, all bridges on the same Compute node should be of type **ovs_user_bridge**. The director may accept the configuration, but Red Hat OpenStack Platform does not support mixing **ovs_bridge** and **ovs_user_bridge** on the same node.

9.6. DEPLOYING THE OVERCLOUD

The following example defines the **openstack overcloud deploy** Bash script that uses composable roles:

```

#!/bin/bash

openstack overcloud deploy \
--templates \
-r /home/stack/ospd-12-sriov-dpdk-heterogeneous-cluster/roles_data.yaml \
-e /usr/share/openstack-tripleo-heat-templates/environments/network-
isolation.yaml \
-e /usr/share/openstack-tripleo-heat-templates/environments/host-config-
and-reboot.yaml \
-e /usr/share/openstack-tripleo-heat-templates/environments/neutron-
sriov.yaml \
-e /usr/share/openstack-tripleo-heat-templates/environments/neutron-ovs-
dpdk.yaml \
-e /home/stack/ospd-12-vlan-sriov-two-ports-ctlplane-bonding/docker-
images.yaml \
-e /home/stack/ospd-12-sriov-dpdk-heterogeneous-cluster/network-
environment.yaml

```

Where:

- `/home/stack/ospd-12-sriov-dpdk-heterogeneous-cluster/roles_data.yaml` is the location of the updated `roles_data.yaml` file, which defines the OVS-DPDK and the SR-IOV composable roles.
- `/home/stack/ospd-12-sriov-dpdk-heterogeneous-cluster/network-environment.yaml` contains the role-specific parameters for the SR-IOV Compute node and the OVS-DPDK Compute node.

CHAPTER 10. CONFIGURING SR-IOV AND DPDK INTERFACES ON THE SAME COMPUTE NODE

This section describes how to deploy SR-IOV and DPDK interfaces on the same Compute node. This deployment uses a custom role for OVS-DPDK and SR-IOV with role-specific parameters defined in the `network-environment.yaml` file. The process to create and deploy a custom role includes:

- Define the custom role to support SR-IOV and DPDK interfaces on the same Compute node.
- Set the role-specific parameters for SR-IOV role and OVS-DPDK in the `network-environment.yaml` file.
- Configure the `compute.yaml` file with an SR-IOV interface and a DPDK interface.
- Deploy the overcloud with this updated set of roles.
- Create the appropriate OpenStack flavor, networks, and ports to support these interface types.

You must install and configure the undercloud before you can deploy the compute node in the overcloud. See the [Director Installation and Usage Guide](#) for details.



NOTE

Ensure that you create an OpenStack flavor that match this custom role.



IMPORTANT

You must determine the best values for the OVS-DPDK parameters that you set in the `network-environment.yaml` file to optimize your OpenStack network for OVS-DPDK. See [Section 8.1, “Deriving DPDK parameters with workflows”](#) for details.

In this example, the `ComputeOvsDpdkSriov` is a custom role for compute nodes to enable DPDK and SR-IOV only on the nodes that have the appropriate NICs. The existing set of default roles provided by Red Hat OpenStack Platform is stored in the `/usr/share/openstack-tripleo-heat-templates/roles_data.yaml` file.

10.1. CREATING THE COMPUTEOVSDPDKSRIOV COMPOSABLE ROLE

Red Hat OpenStack Platform provides a set of default roles in the `roles_data.yaml` file. You can create your own `roles_data.yaml` file to support the roles you need.

To create the `ComputeOvsDpdkSriov` composable role to support SR-IOV and DPDK interfaces on the same Compute node:

1. Create the `ComputeOvsDpdkSriov.yaml` file in a local directory and add the definition of this role:

```
#####
#####
# Role: ComputeOvsDpdkSriov
#
#####
#####
```

```

- name: ComputeOvsDpdkSriov
  description: |
    Compute OVS DPDK Sriov Role
  CountDefault: 1
  networks:
    - InternalApi
    - Tenant
    - Storage
  HostnameFormatDefault: 'computeovsdpdkSriov-%index%'
  disable_upgrade_deployment: True
  ServicesDefault:
    - OS::TripleO::Services::AuditD
    - OS::TripleO::Services::CACerts
    - OS::TripleO::Services::CephClient
    - OS::TripleO::Services::CephExternal
    - OS::TripleO::Services::CertmongerUser
    - OS::TripleO::Services::Collectd
    - OS::TripleO::Services::ComputeCeilometerAgent
    - OS::TripleO::Services::ComputeNeutronCorePlugin
    - OS::TripleO::Services::ComputeNeutronL3Agent
    - OS::TripleO::Services::ComputeNeutronMetadataAgent
    - OS::TripleO::Services::ComputeNeutronOvsDpdk
    - OS::TripleO::Services::Docker
    - OS::TripleO::Services::FluentdClient
    - OS::TripleO::Services::Iscsid
    - OS::TripleO::Services::Kernel
    - OS::TripleO::Services::MySQLClient
    - OS::TripleO::Services::NovaCompute
    - OS::TripleO::Services::NovaLibvirt
    - OS::TripleO::Services::NovaMigrationTarget
    - OS::TripleO::Services::NeutronLinuxbridgeAgent
    - OS::TripleO::Services::NeutronSriovAgent
    - OS::TripleO::Services::NeutronVppAgent
    - OS::TripleO::Services::Ntp
    - OS::TripleO::Services::ContainersLogrotateCronD
    - OS::TripleO::Services::OpenDaylightOvs
    - OS::TripleO::Services::Securetty
    - OS::TripleO::Services::SensuClient
    - OS::TripleO::Services::Snmp
    - OS::TripleO::Services::Sshd
    - OS::TripleO::Services::Timezone
    - OS::TripleO::Services::TripleoFirewall
    - OS::TripleO::Services::TripleoPackages
    - OS::TripleO::Services::Tuned
    - OS::TripleO::Services::Vpp
    - OS::TripleO::Services::OVNController

```

2. Generate `roles_data.yaml` for the **ComputeOvsDpdkSriov** role and any other roles you need for your deployment:

```

# openstack overcloud roles generate --roles-path
templates/openstack-tripleo-heat-templates/roles -o roles_data.yaml
Controller ComputeOvsDpdkSriov

```

10.2. CONFIGURING TUNED FOR CPU AFFINITY

This example uses the sample [post-install.yaml](#) file.

1. Set the tuned configuration to enable CPU affinity.

```
ExtraDeployments:
  type: OS::Heat::StructuredDeployments
  properties:
    servers: {get_param: servers}
    config: {get_resource: ExtraConfig}
    actions: ['CREATE', 'UPDATE']

ExtraConfig:
  type: OS::Heat::SoftwareConfig
  properties:
    group: script
    config: |
      #!/bin/bash
      set -x
      function tuned_service_dependency() {
        tuned_service=/usr/lib/systemd/system/tuned.service
        grep -q "network.target" $tuned_service
        if [ "$?" -eq 0 ]; then
          sed -i '/After=.*s/network.target//g' $tuned_service
        fi
        grep -q "Before=.*network.target" $tuned_service
        if [ ! "$?" -eq 0 ]; then
          grep -q "Before=.*" $tuned_service
          if [ "$?" -eq 0 ]; then
            sed -i 's/^\(Before=.*\)\/\1 network.target
openvswitch.service/g' $tuned_service
          else
            sed -i '/After/i Before=network.target
openvswitch.service' $tuned_service
          fi
        fi
      }

      if hiera -c /etc/puppet/hiera.yaml service_names | grep -q
neutron_ovs_dpdk_agent; then
        tuned_service_dependency
      fi
```

10.3. DEFINING THE SR-IOV AND OVS-DPDK ROLE-SPECIFIC PARAMETERS

Modify the [network-environment.yaml](#) file to configure SR-IOV and OVS-DPDK role-specific parameters:

1. Add the resource mapping for the OVS-DPDK and SR-IOV services to the `network-environment.yaml` file along with the network configuration for these nodes:

```
resource_registry:
  # Specify the relative/absolute path to the config files you
  # want to use for override the default.
  OS::TripleO::ComputeOvsDpdkSriov::Net::SoftwareConfig: nic-
```



```

configs/computeovsdpdksriv.yaml
  OS::TripleO::Controller::Net::SoftwareConfig: nic-
configs/controller.yaml
  OS::TripleO::NodeExtraConfigPost: post-install.yaml

```

2. Define the flavors:

```

OvercloudControllerFlavor: controller
OvercloudComputeOvsDpdkSrivFlavor: compute

```

3. Configure the role-specific parameters for SR-IOV:

```

#SR-IOV params
NeutronMechanismDrivers: ['openvswitch','sriovnicswitch']
NovaSchedulerDefaultFilters:
['RetryFilter','AvailabilityZoneFilter','RamFilter','ComputeFilter',
'ComputeCapabilitiesFilter','ImagePropertiesFilter','ServerGroupAnti
AffinityFilter','ServerGroupAffinityFilter','PciPassthroughFilter','
NUMATopologyFilter']
NovaSchedulerAvailableFilters:
['nova.scheduler.filters.all_filters',"nova.scheduler.filters.pci_pa
ssthrough_filter.PciPassthroughFilter"]
NeutronSupportedPCIVendorDevs: ['8086:154d', '8086:10ed']
NovaPCIPassthrough:
- devname: "ens2f1"
  physical_network: "tenant"

NeutronPhysicalDevMappings: "tenant:ens2f1"
NeutronSrivNumVFs: "ens2f1:5"

```

4. Configure the role-specific parameters for OVS-DPDK:

```

#####
# OVS DPDK configuration #
# #####
ComputeOvsDpdkSrivParameters:
  KernelArgs: default_hugepagesz=1GB hugepagesz=1G hugepages=32
iommu=pt intel_iommu=on
  TunedProfileName: "cpu-partitioning"
  IsolCpusList:
"1,2,3,4,5,6,7,9,10,17,18,19,20,21,22,23,11,12,13,14,15,25,26,27,28,
29,30,31"
  NovaVcpuPinSet:
['2,3,4,5,6,7,18,19,20,21,22,23,10,11,12,13,14,15,26,27,28,29,30,31'
]
  NovaReservedHostMemory: 4096
  OvsDpdkSocketMemory: "1024,1024"
  OvsDpdkMemoryChannels: "4"
  OvsDpdkCoreList: "0,16,8,24"
  OvsPmdCoreList: "1,17,9,25"

# MTU global configuration
NeutronGlobalPhysnetMtu: 9000
# DHCP provide metadata route to VM.
NeutronEnableIsolatedMetadata: true

```

```
# DHCP always provides metadata route to VM.
NeutronEnableForceMetadata: true
# Configure the classname of the firewall driver to use for
implementing security groups.
NeutronOVSEthernetDriver: openvswitch
```



NOTE

You must assign at least one CPU (with sibling thread) on each NUMA node with or without DPDK NICs present for DPDK PMD to avoid failures in creating guest instances.

5. Configure the remainder of the `network-environment.yaml` file to override the default parameters from the `neutron-ovs-dpdk-agent.yaml` and `neutron-sriov-agent.yaml` files as needed for your OpenStack deployment.

See [Planning your OVS-DPDK Deployment](#) for details on how to determine the best values for the OVS-DPDK parameters that you set in the `network-environment.yaml` file to optimize your OpenStack network for OVS-DPDK.

10.4. CONFIGURING THE COMPUTE NODE FOR SR-IOV AND DPDK INTERFACES

This example uses the sample the [compute.yaml](#) file to create the `computeovsdpdk-sriov.yaml` file to support SR-IOV and DPDK interfaces.

1. Create the control plane Linux bond for an isolated network:

```
- type: linux_bond
  name: bond_api
  bonding_options: "mode=active-backup"
  use_dhcp: false
  dns_servers:
    get_param: DnsServers
  members:
    - type: interface
      name: nic3
      primary: true
    - type: interface
      name: nic4
```

2. Assign VLANs to this Linux bond:

```
- type: vlan
  vlan_id:
    get_param: InternalApiNetworkVlanID
  device: bond_api
  addresses:
    - ip_netmask:
        get_param: InternalApiIpSubnet

- type: vlan
  vlan_id:
    get_param: TenantNetworkVlanID
```

```

device: bond_api
addresses:
- ip_netmask:
    get_param: TenantIpSubnet

- type: vlan
  vlan_id:
    get_param: StorageNetworkVlanID
  device: bond_api
  addresses:
  - ip_netmask:
      get_param: StorageIpSubnet

```

3. Set a bridge with a DPDK port to link to the controller:

```

- type: ovs_user_bridge
  name: br-link0
  use_dhcp: false
  members:
  - type: ovs_dpdk_port
    name: dpdk0
    mtu: 9000
    rx_queue: 2
    members:
    - type: interface
      name: nic5

```



NOTE

To include multiple DPDK devices, repeat the **type** code section for each DPDK device you want to add.



NOTE

When using OVS-DPDK, all bridges on the same Compute node should be of type **ovs_user_bridge**. The director may accept the configuration, but Red Hat OpenStack Platform does not support mixing **ovs_bridge** and **ovs_user_bridge** on the same node.

4. Create the SR-IOV interface to the Controller:

```

- type: interface
  name: ens2f1
  mtu: 9000
  use_dhcp: false
  defroute: false
  nm_controlled: true
  hotplug: true

```

10.5. DEPLOYING THE OVERCLOUD

The following example defines the **openstack overcloud deploy** Bash script that uses composable roles:

```
#!/bin/bash
```

```
openstack overcloud deploy \
--templates \
-r /home/stack/ospd-12-vlan-dpdk-sriov-two-ports-ctlplane-
bonding/roles_data.yaml \
-e /usr/share/openstack-tripleo-heat-templates/environments/network-
isolation.yaml \
-e /usr/share/openstack-tripleo-heat-templates/environments/host-config-
and-reboot.yaml \
-e /usr/share/openstack-tripleo-heat-templates/environments/neutron-ovs-
dpdk.yaml \
-e /usr/share/openstack-tripleo-heat-templates/environments/neutron-
sriov.yaml \
-e /home/stack/ospd-12-vlan-dpdk-sriov-two-ports-ctlplane-bonding/docker-
images.yaml \
-e /home/stack/ospd-12-vlan-dpdk-sriov-two-ports-ctlplane-bonding/network-
environment.yaml
```

Where:

- **/home/stack/ospd-12-vlan-dpdk-sriov-two-ports-ctlplane-bonding/roles_data.yaml** is the location of the updated **roles_data.yaml** file, which defines the **ComputeOvsDpdkSriov** custom role.
- **/home/stack/ospd-12-vlan-dpdk-sriov-two-ports-ctlplane-bonding/network-environment.yaml** contains the role-specific parameters for SR-IOV OVS-DPDK interfaces.

10.6. CREATING A FLAVOR AND DEPLOYING AN INSTANCE WITH SR-IOV AND DPDK INTERFACES

After you have completed configuring SR-IOV and DPDK interfaces on the same compute node, you need to create a flavor and deploy an instance by performing the following steps:



NOTE

You should use host aggregates to separate CPU pinned instances from unpinned instances. Instances that do not use CPU pinning do not respect the resourcing requirements of instances that use CPU pinning.

1. Create a flavor:

```
# openstack flavor create --vcpus 6 --ram 4096 --disk 40 compute
```

Where:

- **compute** is the flavor name.
- **4096** is the memory size in MB.
- **40** is the disk size in GB (default 0G).
- **6** is the number of vCPUs.

2. Set the flavor for large pages:

```
# openstack flavor set compute --property hw:mem_page_size=1GB
```

3. Create the networks for SR-IOV and DPDK:

```
# openstack network create --name net-dpdk
# openstack network create --name net-sriov
# openstack subnet create --subnet-range <cidr/prefix> --network
net-dpdk --gateway <gateway> net-dpdk-subnet
# openstack subnet create --subnet-range <cidr/prefix> --network
net-sriov --gateway <gateway> net-sriov-subnet
```

4. Create the SR-IOV port.

a. Use **vnic-type direct** to create an SR-IOV VF port:

```
# openstack port create --network net-sriov --vnic-type direct
sriov_port
```

b. Use **vnic-type direct-physical** to create an SR-IOV PF port:

```
# openstack port create --network net-sriov --vnic-type direct-
physical sriov_port
```

5. Deploy an instance:

```
# openstack server create --flavor compute --image rhel_7.3 --nic
port-id=sriov_port --nic net-id=net-dpdk vm1
```

Where:

- **compute** is the flavor name or ID.
- **rhel_7.3** is the image (name or ID) used to create an instance.
- **sriov_port** is the SR-IOV NIC on the Compute node.
- **net-dpdk** is the DPDK network.
- **vm1** is the name of the instance.

You have now deployed an instance that uses an SR-IOV interface and a DPDK interface on the same Compute node.

CHAPTER 11. UPGRADING RED HAT OPENSTACK PLATFORM WITH NFV

There are additional considerations and steps needed to upgrade Red Hat OpenStack Platform when you have OVS-DPDK configured. The steps are covered in [Preparing an NFV-Configured Overcloud](#).

CHAPTER 12. PERFORMANCE

Red Hat OpenStack Platform director configures the Compute nodes to enforce resource partitioning and fine tuning to achieve line rate performance for the guest VNFs. The key performance factors in the NFV use case are throughput, latency and jitter.

DPDK-accelerated OVS enables high performance packet switching between physical NICs and virtual machines. OVS 2.7 embeds support for DPDK 16.11 and includes support for `vhost-user` multiqueue, allowing scalable performance. OVS-DPDK provides line rate performance for guest VNFs.

SR-IOV networking provides enhanced performance characteristics, including improved throughput for specific networks and virtual machines.

Other important features for performance tuning include huge pages, NUMA alignment, host isolation and CPU pinning. VNF flavors require huge pages and emulator thread isolation for better performance. Host isolation and CPU pinning improve NFV performance and prevent spurious packet loss.

See [NFV Performance Considerations](#) and [Configure Emulator Threads to run on a Dedicated Physical CPU](#) for a high-level introduction to CPUs and NUMA topology.

CHAPTER 13. FINDING MORE INFORMATION

The following table includes additional Red Hat documentation for reference:

The Red Hat OpenStack Platform documentation suite can be found here: [Red Hat OpenStack Platform Documentation Suite](#)

Table 13.1. List of Available Documentation

Component	Reference
Red Hat Enterprise Linux	Red Hat OpenStack Platform is supported on Red Hat Enterprise Linux 7.4. For information on installing Red Hat Enterprise Linux, see the corresponding installation guide at: Red Hat Enterprise Linux Documentation Suite .
Red Hat OpenStack Platform	<p>To install OpenStack components and their dependencies, use the Red Hat OpenStack Platform director. The director uses a basic OpenStack installation as the undercloud to install, configure and manage the OpenStack nodes in the final overcloud. Be aware that you will need one extra host machine for the installation of the undercloud, in addition to the environment necessary for the deployed overcloud. For detailed instructions, see Red Hat OpenStack Platform Director Installation and Usage.</p> <p>For information on configuring advanced features for a Red Hat OpenStack Platform enterprise environment using the Red Hat OpenStack Platform director such as network isolation, storage configuration, SSL communication, and general configuration method, see Advanced Overcloud Customization.</p>
NFV Documentation	For a high level overview of the NFV concepts, see the Network Functions Virtualization Product Guide .

APPENDIX A. SAMPLE SR-IOV YAML FILES

This section provides sample SR-IOV YAML files as a reference.

A.1. SAMPLE SR-IOV TWO-PORT CONTROL PLANE BONDING YAML FILES

A.1.1. roles_data.yaml

```
#####
#####
# File generated by TripleO
#####
#####
#####
#####
# Role: Controller
#
#####
#####
- name: Controller
  description: |
    Controller role that has all the controller services loaded and handles
    Database, Messaging and Network functions.
  CountDefault: 1
  tags:
    - primary
    - controller
  networks:
    - External
    - InternalApi
    - Storage
    - StorageMgmt
    - Tenant
  HostnameFormatDefault: 'controller-%index%'
  # Deprecated & backward-compatible values (FIXME: Make parameters
consistent)
  # Set uses_deprecated_params to True if any deprecated params are used.
  uses_deprecated_params: True
  deprecated_param_extraconfig: 'controllerExtraConfig'
  deprecated_param_flavor: 'OvercloudControlFlavor'
  deprecated_param_image: 'controllerImage'
  ServicesDefault:
    - OS::TripleO::Services::AodhApi
    - OS::TripleO::Services::AodhEvaluator
    - OS::TripleO::Services::AodhListener
    - OS::TripleO::Services::AodhNotifier
    - OS::TripleO::Services::AuditD
    - OS::TripleO::Services::BarbicanApi
    - OS::TripleO::Services::CACerts
    - OS::TripleO::Services::CeilometerAgentCentral
    - OS::TripleO::Services::CeilometerAgentNotification
    # FIXME: This service was disabled in Pike and this entry should be
removed
```

```
# in Queens.
- OS::Triple0::Services::CeilometerApi
- OS::Triple0::Services::CeilometerCollector
- OS::Triple0::Services::CeilometerExpirer
- OS::Triple0::Services::CephExternal
- OS::Triple0::Services::CephMds
- OS::Triple0::Services::CephMon
- OS::Triple0::Services::CephRbdMirror
- OS::Triple0::Services::CephRgw
- OS::Triple0::Services::CertmongerUser
- OS::Triple0::Services::CinderApi
- OS::Triple0::Services::CinderBackendDellPs
- OS::Triple0::Services::CinderBackendDellSc
- OS::Triple0::Services::CinderBackendDellEMCUnity
- OS::Triple0::Services::CinderBackendDellEMCVMAXISCSI
- OS::Triple0::Services::CinderBackendNetApp
- OS::Triple0::Services::CinderBackendScaleIO
- OS::Triple0::Services::CinderBackendVRTSHyperScale
- OS::Triple0::Services::CinderBackup
- OS::Triple0::Services::CinderHPELeftHandISCSI
- OS::Triple0::Services::CinderScheduler
- OS::Triple0::Services::CinderVolume
- OS::Triple0::Services::Clustercheck
- OS::Triple0::Services::Collectd
- OS::Triple0::Services::Congress
- OS::Triple0::Services::Docker
- OS::Triple0::Services::Ec2Api
- OS::Triple0::Services::Etcd
- OS::Triple0::Services::ExternalSwiftProxy
- OS::Triple0::Services::FluentdClient
- OS::Triple0::Services::GlanceApi
- OS::Triple0::Services::GnocchiApi
- OS::Triple0::Services::GnocchiMetricd
- OS::Triple0::Services::GnocchiStatsd
- OS::Triple0::Services::HAproxy
- OS::Triple0::Services::HeatApi
- OS::Triple0::Services::HeatApiCfn
- OS::Triple0::Services::HeatApiCloudwatch
- OS::Triple0::Services::HeatEngine
- OS::Triple0::Services::Horizon
- OS::Triple0::Services::IronicApi
- OS::Triple0::Services::IronicConductor
- OS::Triple0::Services::Iscsid
- OS::Triple0::Services::Keepalived
- OS::Triple0::Services::Kernel
- OS::Triple0::Services::Keystone
- OS::Triple0::Services::ManilaApi
- OS::Triple0::Services::ManilaBackendCephFs
- OS::Triple0::Services::ManilaBackendGeneric
- OS::Triple0::Services::ManilaBackendIsilon
- OS::Triple0::Services::ManilaBackendNetapp
- OS::Triple0::Services::ManilaBackendUnity
- OS::Triple0::Services::ManilaBackendVNX
- OS::Triple0::Services::ManilaBackendVMAX
- OS::Triple0::Services::ManilaScheduler
- OS::Triple0::Services::ManilaShare
```

- OS::Triple0::Services::Memcached
- OS::Triple0::Services::MongoDb
- OS::Triple0::Services::MySQL
- OS::Triple0::Services::MySQLClient
- OS::Triple0::Services::NeutronApi
- OS::Triple0::Services::NeutronBgpVpnApi
- OS::Triple0::Services::NeutronCorePlugin
- OS::Triple0::Services::NeutronDhcpAgent
- OS::Triple0::Services::NeutronL2gwAgent
- OS::Triple0::Services::NeutronL2gwApi
- OS::Triple0::Services::NeutronL3Agent
- OS::Triple0::Services::NeutronLbaasv2Agent
- OS::Triple0::Services::NeutronLinuxbridgeAgent
- OS::Triple0::Services::NeutronMetadataAgent
- OS::Triple0::Services::NeutronML2FujitsuCfab
- OS::Triple0::Services::NeutronML2FujitsuFossw
- OS::Triple0::Services::NeutronOvsAgent
- OS::Triple0::Services::NeutronVppAgent
- OS::Triple0::Services::NovaApi
- OS::Triple0::Services::NovaConductor
- OS::Triple0::Services::NovaConsoleauth
- OS::Triple0::Services::NovaIronic
- OS::Triple0::Services::NovaMetadata
- OS::Triple0::Services::NovaPlacement
- OS::Triple0::Services::NovaScheduler
- OS::Triple0::Services::NovaVncProxy
- OS::Triple0::Services::Ntp
- OS::Triple0::Services::ContainersLogrotateCronD
- OS::Triple0::Services::OctaviaApi
- OS::Triple0::Services::OctaviaHealthManager
- OS::Triple0::Services::OctaviaHousekeeping
- OS::Triple0::Services::OctaviaWorker
- OS::Triple0::Services::OpenDaylightApi
- OS::Triple0::Services::OpenDaylightOvs
- OS::Triple0::Services::OVNDBs
- OS::Triple0::Services::OVNController
- OS::Triple0::Services::Pacemaker
- OS::Triple0::Services::PankoApi
- OS::Triple0::Services::RabbitMQ
- OS::Triple0::Services::Redis
- OS::Triple0::Services::SaharaApi
- OS::Triple0::Services::SaharaEngine
- OS::Triple0::Services::Securetty
- OS::Triple0::Services::SensuClient
- OS::Triple0::Services::Snmp
- OS::Triple0::Services::Sshd
- OS::Triple0::Services::SwiftProxy
- OS::Triple0::Services::SwiftRingBuilder
- OS::Triple0::Services::SwiftStorage
- OS::Triple0::Services::Tacker
- OS::Triple0::Services::Timezone
- OS::Triple0::Services::TripleoFirewall
- OS::Triple0::Services::TripleoPackages
- OS::Triple0::Services::Tuned
- OS::Triple0::Services::Vpp
- OS::Triple0::Services::Zaqar

```
#####
####
# Role: Compute
#
#####
####
- name: ComputeSriov
  description: |
    Compute Sriov Node role
  CountDefault: 1
  networks:
    - InternalApi
    - Tenant
    - Storage
  HostnameFormatDefault: 'computesriov-%index%'
  # Deprecated & backward-compatible values (FIXME: Make parameters
consistent)
  # Set uses_deprecated_params to True if any deprecated params are used.
  uses_deprecated_params: True
  deprecated_param_image: 'NovaImage'
  deprecated_param_extraconfig: 'NovaComputeExtraConfig'
  deprecated_param_metadata: 'NovaComputeServerMetadata'
  deprecated_param_scheduler_hints: 'NovaComputeSchedulerHints'
  deprecated_param_ips: 'NovaComputeIPs'
  deprecated_server_resource_name: 'NovaCompute'
  disable_upgrade_deployment: True
  ServicesDefault:
    - OS::Triple0::Services::AuditD
    - OS::Triple0::Services::CACerts
    - OS::Triple0::Services::CephClient
    - OS::Triple0::Services::CephExternal
    - OS::Triple0::Services::CertmongerUser
    - OS::Triple0::Services::Collectd
    - OS::Triple0::Services::ComputeCeilometerAgent
    - OS::Triple0::Services::ComputeNeutronCorePlugin
    - OS::Triple0::Services::ComputeNeutronL3Agent
    - OS::Triple0::Services::ComputeNeutronMetadataAgent
    - OS::Triple0::Services::ComputeNeutronOvsAgent
    - OS::Triple0::Services::Docker
    - OS::Triple0::Services::FluentdClient
    - OS::Triple0::Services::Iscsid
    - OS::Triple0::Services::Kernel
    - OS::Triple0::Services::MySQLClient
    - OS::Triple0::Services::NeutronLinuxbridgeAgent
    - OS::Triple0::Services::NeutronSriovAgent
    - OS::Triple0::Services::NeutronSriovHostConfig
    - OS::Triple0::Services::NeutronVppAgent
    - OS::Triple0::Services::NovaCompute
    - OS::Triple0::Services::NovaLibvirt
    - OS::Triple0::Services::NovaMigrationTarget
    - OS::Triple0::Services::Ntp
    - OS::Triple0::Services::ContainersLogrotateCronD
    - OS::Triple0::Services::OpenDaylightOvs
    - OS::Triple0::Services::Securetty
    - OS::Triple0::Services::SensuClient
    - OS::Triple0::Services::Snmp
```

- OS::TripleO::Services::Sshd
- OS::TripleO::Services::Timezone
- OS::TripleO::Services::TripleoFirewall
- OS::TripleO::Services::TripleoPackages
- OS::TripleO::Services::Tuned
- OS::TripleO::Services::Vpp
- OS::TripleO::Services::OVNController

A.1.2. post-install.yaml

```

heat_template_version: 2014-10-16

description: >
  Example extra config for post-deployment

parameters:
  servers:
    type: json

resources:

  ExtraDeployments:
    type: OS::Heat::StructuredDeployments
    properties:
      servers: {get_param: servers}
      config: {get_resource: ExtraConfig}
      actions: ['CREATE', 'UPDATE']

  ExtraConfig:
    type: OS::Heat::SoftwareConfig
    properties:
      group: script
      config: |
        #!/bin/bash
        set -x
        function tuned_service_dependency() {
          tuned_service=/usr/lib/systemd/system/tuned.service
          grep -q "network.target" $tuned_service
          if [ "$?" -eq 0 ]; then
            sed -i '/After=.*s/network.target//g' $tuned_service
          fi
          grep -q "Before=.*network.target" $tuned_service
          if [ ! "$?" -eq 0 ]; then
            grep -q "Before=.*" $tuned_service
            if [ "$?" -eq 0 ]; then
              sed -i 's/^(Before=.*\)/\1 network.target
openvswitch.service/g' $tuned_service
            else
              sed -i '/After/i Before=network.target
openvswitch.service' $tuned_service
            fi
          fi
        }

        if hiera -c /etc/puppet/hiera.yaml service_names | grep -q

```

```

neutron_ovs_dpdk_agent; then
    tuned_service_dependency
fi

```

A.1.3. network.environment.yaml

```

resource_registry:
    # Specify the relative/absolute path to the config files you want to use
    # for override the default.
    OS::TripleO::ComputeSriov::Net::SoftwareConfig: nic-configs/compute-
sriov.yaml
    OS::TripleO::Controller::Net::SoftwareConfig: nic-
configs/controller.yaml
    OS::TripleO::NodeExtraConfigPost: post-install.yaml

parameter_defaults:
    # Customize all these values to match the local environment
    InternalApiNetCidr: 10.10.10.0/24
    TenantNetCidr: 10.10.2.0/24
    StorageNetCidr: 10.10.3.0/24
    StorageMgmtNetCidr: 10.10.4.0/24
    ExternalNetCidr: 172.20.12.112/28
    # CIDR subnet mask length for provisioning network
    ControlPlaneSubnetCidr: '24'
    InternalApiAllocationPools: [{'start': '10.10.10.10', 'end':
'10.10.10.200'}]
    TenantAllocationPools: [{'start': '10.10.2.100', 'end': '10.10.2.200'}]
    StorageAllocationPools: [{'start': '10.10.3.100', 'end': '10.10.3.200'}]
    StorageMgmtAllocationPools: [{'start': '10.10.4.100', 'end':
'10.10.4.200'}]
    # Use an External allocation pool which will leave room for floating IPs
    ExternalAllocationPools: [{'start': '172.20.12.114', 'end':
'172.20.12.125'}]
    # Set to the router gateway on the external network
    ExternalInterfaceDefaultRoute: 172.20.12.126
    # Gateway router for the provisioning network (or Undercloud IP)
    ControlPlaneDefaultRoute: 192.168.24.1
    # Generally the IP of the Undercloud
    EC2MetadataIp: 192.168.24.1
    InternalApiNetworkVlanID: 10
    TenantNetworkVlanID: 11
    StorageNetworkVlanID: 12
    StorageMgmtNetworkVlanID: 13
    ExternalNetworkVlanID: 14
    # Define the DNS servers (maximum 2) for the overcloud nodes
    DnsServers: ["8.8.8.8", "8.8.4.4"]
    # May set to br-ex if using floating IPs only on native VLAN on bridge
br-ex
    NeutronExternalNetworkBridge: ""
    # The tunnel type for the tenant network (vxlan or gre). Set to '' to
disable tunneling.
    NeutronTunnelTypes: ''
    # The tenant network type for Neutron (vlan or vxlan).
    NeutronNetworkType: 'vlan'
    # The OVS logical->physical bridge mappings to use.

```

```

NeutronBridgeMappings: 'tenant:br-link0'
# The Neutron ML2 and OpenVSwitch vlan mapping range to support.
NeutronNetworkVLANRanges: 'tenant:22:22,tenant:25:25'
# Nova flavor to use.
OvercloudControllerFlavor: controller
OvercloudComputeSriovFlavor: compute
#Number of nodes to deploy.
ControllerCount: 1
ComputeSriovCount: 1
# NTP server configuration.
NtpServer: clock.redhat.com

#####
# SRIOV configuration #
#####
# The mechanism drivers for the Neutron tenant network.
NeutronMechanismDrivers: "openvswitch,sriovnicswitch"
# List of PCI Passthrough whitelist parameters.
# Use ONE of the following examples.
# Example 1:
# NovaPCIPassthrough:
#   - vendor_id: "8086"
#     product_id: "154c"
#     address: "0000:05:00.0" - (optional)
#     physical_network: "datacentre"
#
# Example 2:
# NovaPCIPassthrough:
#   - devname: "p6p1"
#     physical_network: "tenant"
NovaPCIPassthrough:
  - devname: "ens2f0"
    physical_network: "tenant"
  - devname: "ens2f1"
    physical_network: "tenant"
# List of supported pci vendor devices in the format VendorID:ProductID.
NeutronSupportedPCIVendorDevs: ['8086:154d', '8086:10ed']
# List of <physical_network>:<physical device>
# All physical networks listed in network_vlan_ranges on the server
# should have mappings to appropriate interfaces on each agent.
NeutronPhysicalDevMappings: "tenant:ens2f0,tenant:ens2f1"
# Provide the list of VFs to be reserved for each SR-IOV interface.
# Format "<interface_name1>:<numvfs1>", "<interface_name2>:<numvfs2>"
# Example "eth1:4096", "eth2:128"
NeutronSriovNumVFs: "ens2f0:5,ens2f1:5"

#####
# Additional computes configuration #
#####

# SR-IOV compute node.
ComputeSriovParameters:
  KernelArgs: default_hugepagesz=1GB hugepagesz=1G hugepages=32 iommu=pt
intel_iommu=on
  TunedProfileName: "cpu-partitioning"
  IsolCpusList:

```

```

"1,2,3,4,5,6,7,9,10,17,18,19,20,21,22,23,11,12,13,14,15,25,26,27,28,29,30,
31"
    NovaVcpuPinSet:
['1,2,3,4,5,6,7,9,10,17,18,19,20,21,22,23,11,12,13,14,15,25,26,27,28,29,30
,31']
    NovaReservedHostMemory: 4096

# MTU global configuration
NeutronGlobalPhysnetMtu: 9000
# DHCP provide metadata route to VM.
NeutronEnableIsolatedMetadata: true
# DHCP always provides metadata route to VM.
NeutronEnableForceMetadata: true
# Configure the classname of the firewall driver to use for implementing
security groups.
NeutronOVSFirewallDriver: openvswitch

# List of scheduler available filters
NovaSchedulerAvailableFilters:
["nova.scheduler.filters.all_filters","nova.scheduler.filters.pci_passthro
ugh_filter.PciPassthroughFilter"]
# An array of filters used by Nova to filter a node. These filters will
be applied in the order they are listed,
# so place your most restrictive filters first to make the filtering
process more efficient.
NovaSchedulerDefaultFilters:
['AvailabilityZoneFilter','RamFilter','ComputeFilter','ComputeCapabilities
Filter','ImagePropertiesFilter','ServerGroupAntiAffinityFilter','ServerGro
upAffinityFilter','PciPassthroughFilter']

```

A.1.4. controller.yaml

```

heat_template_version: pike

description: >
  Software Config to drive os-net-config to configure VLANs for the
  controller role.

parameters:
  ControlPlaneIp:
    default: ''
    description: IP address/subnet on the ctlplane network
    type: string
  ExternalIpSubnet:
    default: ''
    description: IP address/subnet on the external network
    type: string
  InternalApiIpSubnet:
    default: ''
    description: IP address/subnet on the internal API network
    type: string
  StorageIpSubnet:
    default: ''
    description: IP address/subnet on the storage network
    type: string

```



```

StorageMgmtIpSubnet:
  default: ''
  description: IP address/subnet on the storage mgmt network
  type: string
StorageNetworkVlanID:
  default: 30
  description: Vlan ID for the storage network traffic.
  type: number
StorageMgmtNetworkVlanID:
  default: 40
  description: Vlan ID for the storage mgmt network traffic.
  type: number
TenantIpSubnet:
  default: ''
  description: IP address/subnet on the tenant network
  type: string
ManagementIpSubnet: # Only populated when including
environments/network-management.yaml
  default: ''
  description: IP address/subnet on the management network
  type: string
ExternalNetworkVlanID:
  default: ''
  description: Vlan ID for the external network traffic.
  type: number
InternalApiNetworkVlanID:
  default: ''
  description: Vlan ID for the internal_api network traffic.
  type: number
TenantNetworkVlanID:
  default: ''
  description: Vlan ID for the tenant network traffic.
  type: number
ManagementNetworkVlanID:
  default: 23
  description: Vlan ID for the management network traffic.
  type: number
ExternalInterfaceDefaultRoute:
  default: ''
  description: default route for the external network
  type: string
ControlPlaneSubnetCidr: # Override this via parameter_defaults
  default: '24'
  description: The subnet CIDR of the control plane network.
  type: string
DnsServers: # Override this via parameter_defaults
  default: []
  description: A list of DNS servers (2 max for some implementations)
that will be added to resolv.conf.
  type: comma_delimited_list
EC2MetadataIp: # Override this via parameter_defaults
  description: The IP address of the EC2 metadata server.
  type: string

resources:
  OsNetConfigImpl:

```

```

type: OS::Heat::SoftwareConfig
properties:
  group: script
  config:
    str_replace:
      template:
        get_file: /usr/share/openstack-tripleo-heat-
templates/network/scripts/run-os-net-config.sh
    params:
      $network_config:
        network_config:
          - type: interface
            name: nic1
            use_dhcp: false
            defroute: false

          - type: linux_bond
            name: bond_api
            bonding_options: "mode=active-backup"
            use_dhcp: false
            dns_servers:
              get_param: DnsServers
            addresses:
              - ip_netmask:
                  list_join:
                    - /
                    - - get_param: ControlPlaneIp
                      - get_param: ControlPlaneSubnetCidr
            routes:
              - ip_netmask: 169.254.169.254/32
                next_hop:
                  get_param: EC2MetadataIp
            members:
              - type: interface
                name: nic2
                primary: true

          - type: vlan
            vlan_id:
              get_param: InternalApiNetworkVlanID
            device: bond_api
            addresses:
              - ip_netmask:
                  get_param: InternalApiIpSubnet

          - type: vlan
            vlan_id:
              get_param: TenantNetworkVlanID
            device: bond_api
            addresses:
              - ip_netmask:
                  get_param: TenantIpSubnet

          - type: vlan
            vlan_id:
              get_param: StorageNetworkVlanID

```

```

        device: bond_api
        addresses:
        - ip_netmask:
            get_param: StorageIpSubnet

    - type: vlan
      vlan_id:
        get_param: StorageMgmtNetworkVlanID
      device: bond_api
      addresses:
      - ip_netmask:
          get_param: StorageMgmtIpSubnet

    - type: vlan
      vlan_id:
        get_param: ExternalNetworkVlanID
      device: bond_api
      addresses:
      - ip_netmask:
          get_param: ExternalIpSubnet
      routes:
      - default: true
        next_hop:
          get_param: ExternalInterfaceDefaultRoute

    - type: ovs_bridge
      name: br-link0
      use_dhcp: false
      members:
      - type: interface
        name: nic3
        mtu: 9000

outputs:
  OS::stack_id:
    description: The OsNetConfigImpl resource.
    value:
      get_resource: OsNetConfigImpl

```

A.1.5. compute-sriov.yaml

```

heat_template_version: pike

description: >
  Software Config to drive os-net-config to configure VLANs for the
  compute role.

parameters:
  ControlPlaneIp:
    default: ''
    description: IP address/subnet on the ctlplane network
    type: string
  ExternalIpSubnet:
    default: ''
    description: IP address/subnet on the external network

```

```

    type: string
InternalApiIpSubnet:
    default: ''
    description: IP address/subnet on the internal API network
    type: string
TenantIpSubnet:
    default: ''
    description: IP address/subnet on the tenant network
    type: string
ManagementIpSubnet: # Only populated when including
environments/network-management.yaml
    default: ''
    description: IP address/subnet on the management network
    type: string
InternalApiNetworkVlanID:
    default: ''
    description: Vlan ID for the internal_api network traffic.
    type: number
StorageNetworkVlanID:
    default: 30
    description: Vlan ID for the storage network traffic.
    type: number
StorageMgmtNetworkVlanID:
    default: 40
    description: Vlan ID for the storage mgmt network traffic.
    type: number
TenantNetworkVlanID:
    default: ''
    description: Vlan ID for the tenant network traffic.
    type: number
ManagementNetworkVlanID:
    default: 23
    description: Vlan ID for the management network traffic.
    type: number
StorageIpSubnet:
    default: ''
    description: IP address/subnet on the storage network
    type: string
StorageMgmtIpSubnet:
    default: ''
    description: IP address/subnet on the storage mgmt network
    type: string
ControlPlaneSubnetCidr: # Override this via parameter_defaults
    default: '24'
    description: The subnet CIDR of the control plane network.
    type: string
ControlPlaneDefaultRoute: # Override this via parameter_defaults
    description: The default route of the control plane network.
    type: string
DnsServers: # Override this via parameter_defaults
    default: []
    description: A list of DNS servers (2 max for some implementations)
that will be added to resolv.conf.
    type: comma_delimited_list
EC2MetadataIp: # Override this via parameter_defaults
    description: The IP address of the EC2 metadata server.

```

```

    type: string
  ExternalInterfaceDefaultRoute:
    default: ''
    description: default route for the externalheat stack-list network
    type: string

resources:
  OsNetConfigImpl:
    type: OS::Heat::SoftwareConfig
    properties:
      group: script
      config:
        str_replace:
          template:
            get_file: /usr/share/openstack-tripleo-heat-
templates/network/scripts/run-os-net-config.sh
        params:
          $network_config:
            network_config:
              - type: interface
                name: nic1
                use_dhcp: false
                defroute: false

              - type: interface
                name: nic2
                addresses:
                  - ip_netmask:
                      list_join:
                        - /
                      - - get_param: ControlPlaneIp
                        - get_param: ControlPlaneSubnetCidr
                routes:
                  - ip_netmask: 169.254.169.254/32
                    next_hop:
                      get_param: EC2MetadataIp
                  - next_hop:
                      get_param: ControlPlaneDefaultRoute

              - type: linux_bond
                name: bond_api
                bonding_options: "mode=active-backup"
                use_dhcp: false
                dns_servers:
                  get_param: DnsServers
                members:
                  - type: interface
                    name: nic3
                    primary: true
                  - type: interface
                    name: nic4

              - type: vlan
                vlan_id:
                  get_param: InternalApiNetworkVlanID
                device: bond_api

```

```

addresses:
- ip_netmask:
    get_param: InternalApiIpSubnet

- type: vlan
  vlan_id:
    get_param: TenantNetworkVlanID
  device: bond_api
  addresses:
  - ip_netmask:
      get_param: TenantIpSubnet

- type: vlan
  vlan_id:
    get_param: StorageNetworkVlanID
  device: bond_api
  addresses:
  - ip_netmask:
      get_param: StorageIpSubnet

- type: interface
  name: ens2f0
  mtu: 9000
  use_dhcp: false
  defroute: false
  nm_controlled: true
  hotplug: true

- type: interface
  name: ens2f1
  mtu: 9000
  use_dhcp: false
  defroute: false
  nm_controlled: true
  hotplug: true

```

```

outputs:
  OS::stack_id:
    description: The OsNetConfigImpl resource.
    value:
      get_resource: OsNetConfigImpl

```

A.1.6. overcloud_deploy.sh

```

#!/bin/bash

openstack overcloud deploy \
--templates \
-r /home/stack/ospd-12-vlan-sriov-two-ports-ctlplane-
bonding/roles_data.yaml \
-e /usr/share/openstack-tripleo-heat-templates/environments/network-
isolation.yaml \
-e /usr/share/openstack-tripleo-heat-templates/environments/neutron-
sriov.yaml \
-e /usr/share/openstack-tripleo-heat-templates/environments/host-config-

```

```
and-reboot.yaml \  
-e /home/stack/ospd-12-vlan-sriov-two-ports-ctlplane-bonding/docker-  
images.yaml \  
-e /home/stack/ospd-12-vlan-sriov-two-ports-ctlplane-bonding/network-  
environment.yaml \  
--log-file overcloud_install.log &> overcloud_install.log
```

APPENDIX B. SAMPLE OVS-DPDK YAML FILES

This section provides sample OVS-DPDK YAML files as a reference.

B.1. SAMPLE VLAN OVS-DPDK DATA PLANE BONDING YAML FILES

B.1.1. roles_data.yaml

```
#####
#####
# File generated by TripleO
#####
#####
#####
#####
# Role: Controller
#
#####
#####
- name: Controller
  description: |
    Controller role that has all the controller services loaded and handles
    Database, Messaging and Network functions.
  CountDefault: 1
  tags:
    - primary
    - controller
  networks:
    - External
    - InternalApi
    - Storage
    - StorageMgmt
    - Tenant
  HostnameFormatDefault: 'controller-%index%'
  # Deprecated & backward-compatible values (FIXME: Make parameters
consistent)
  # Set uses_deprecated_params to True if any deprecated params are used.
  uses_deprecated_params: True
  deprecated_param_extraconfig: 'controllerExtraConfig'
  deprecated_param_flavor: 'OvercloudControlFlavor'
  deprecated_param_image: 'controllerImage'
  ServicesDefault:
    - OS::TripleO::Services::AodhApi
    - OS::TripleO::Services::AodhEvaluator
    - OS::TripleO::Services::AodhListener
    - OS::TripleO::Services::AodhNotifier
    - OS::TripleO::Services::AuditD
    - OS::TripleO::Services::BarbicanApi
    - OS::TripleO::Services::CACerts
    - OS::TripleO::Services::CeilometerAgentCentral
    - OS::TripleO::Services::CeilometerAgentNotification
    # FIXME: This service was disabled in Pike and this entry should be
removed
    # in Queens.
```


- OS::Triple0::Services::CeilometerApi
- OS::Triple0::Services::CeilometerCollector
- OS::Triple0::Services::CeilometerExpirer
- OS::Triple0::Services::CephExternal
- OS::Triple0::Services::CephMds
- OS::Triple0::Services::CephMon
- OS::Triple0::Services::CephRbdMirror
- OS::Triple0::Services::CephRgw
- OS::Triple0::Services::CertmongerUser
- OS::Triple0::Services::CinderApi
- OS::Triple0::Services::CinderBackendDellPs
- OS::Triple0::Services::CinderBackendDellSc
- OS::Triple0::Services::CinderBackendDellEMCUnity
- OS::Triple0::Services::CinderBackendDellEMCVMAXISCSI
- OS::Triple0::Services::CinderBackendNetApp
- OS::Triple0::Services::CinderBackendScaleIO
- OS::Triple0::Services::CinderBackendVRTSHyperScale
- OS::Triple0::Services::CinderBackup
- OS::Triple0::Services::CinderHPELeftHandISCSI
- OS::Triple0::Services::CinderScheduler
- OS::Triple0::Services::CinderVolume
- OS::Triple0::Services::Clustercheck
- OS::Triple0::Services::Collectd
- OS::Triple0::Services::Congress
- OS::Triple0::Services::Docker
- OS::Triple0::Services::Ec2Api
- OS::Triple0::Services::Etcd
- OS::Triple0::Services::ExternalSwiftProxy
- OS::Triple0::Services::FluentdClient
- OS::Triple0::Services::GlanceApi
- OS::Triple0::Services::GnocchiApi
- OS::Triple0::Services::GnocchiMetricd
- OS::Triple0::Services::GnocchiStatsd
- OS::Triple0::Services::HAproxy
- OS::Triple0::Services::HeatApi
- OS::Triple0::Services::HeatApiCfn
- OS::Triple0::Services::HeatApiCloudwatch
- OS::Triple0::Services::HeatEngine
- OS::Triple0::Services::Horizon
- OS::Triple0::Services::IronicApi
- OS::Triple0::Services::IronicConductor
- OS::Triple0::Services::Iscsid
- OS::Triple0::Services::Keepalived
- OS::Triple0::Services::Kernel
- OS::Triple0::Services::Keystone
- OS::Triple0::Services::ManilaApi
- OS::Triple0::Services::ManilaBackendCephFs
- OS::Triple0::Services::ManilaBackendGeneric
- OS::Triple0::Services::ManilaBackendIsilon
- OS::Triple0::Services::ManilaBackendNetapp
- OS::Triple0::Services::ManilaBackendUnity
- OS::Triple0::Services::ManilaBackendVNX
- OS::Triple0::Services::ManilaBackendVMAX
- OS::Triple0::Services::ManilaScheduler
- OS::Triple0::Services::ManilaShare
- OS::Triple0::Services::Memcached

- OS::Triple0::Services::MongoDb
- OS::Triple0::Services::MySQL
- OS::Triple0::Services::MySQLClient
- OS::Triple0::Services::NeutronApi
- OS::Triple0::Services::NeutronBgpVpnApi
- OS::Triple0::Services::NeutronCorePlugin
- OS::Triple0::Services::NeutronDhcpAgent
- OS::Triple0::Services::NeutronL2gwAgent
- OS::Triple0::Services::NeutronL2gwApi
- OS::Triple0::Services::NeutronL3Agent
- OS::Triple0::Services::NeutronLbaasv2Agent
- OS::Triple0::Services::NeutronLinuxbridgeAgent
- OS::Triple0::Services::NeutronMetadataAgent
- OS::Triple0::Services::NeutronML2FujitsuCfab
- OS::Triple0::Services::NeutronML2FujitsuFossw
- OS::Triple0::Services::NeutronOvsAgent
- OS::Triple0::Services::NeutronVppAgent
- OS::Triple0::Services::NovaApi
- OS::Triple0::Services::NovaConductor
- OS::Triple0::Services::NovaConsoleauth
- OS::Triple0::Services::NovaIronic
- OS::Triple0::Services::NovaMetadata
- OS::Triple0::Services::NovaPlacement
- OS::Triple0::Services::NovaScheduler
- OS::Triple0::Services::NovaVncProxy
- OS::Triple0::Services::Ntp
- OS::Triple0::Services::ContainersLogrotateCron
- OS::Triple0::Services::OctaviaApi
- OS::Triple0::Services::OctaviaHealthManager
- OS::Triple0::Services::OctaviaHousekeeping
- OS::Triple0::Services::OctaviaWorker
- OS::Triple0::Services::OpenDaylightApi
- OS::Triple0::Services::OpenDaylightOvs
- OS::Triple0::Services::OVNDBs
- OS::Triple0::Services::OVNController
- OS::Triple0::Services::Pacemaker
- OS::Triple0::Services::PankoApi
- OS::Triple0::Services::RabbitMQ
- OS::Triple0::Services::Redis
- OS::Triple0::Services::SaharaApi
- OS::Triple0::Services::SaharaEngine
- OS::Triple0::Services::Securetty
- OS::Triple0::Services::SensuClient
- OS::Triple0::Services::Snmp
- OS::Triple0::Services::Sshd
- OS::Triple0::Services::SwiftProxy
- OS::Triple0::Services::SwiftRingBuilder
- OS::Triple0::Services::SwiftStorage
- OS::Triple0::Services::Tacker
- OS::Triple0::Services::Timezone
- OS::Triple0::Services::TripleoFirewall
- OS::Triple0::Services::TripleoPackages
- OS::Triple0::Services::Tuned
- OS::Triple0::Services::Vpp
- OS::Triple0::Services::Zaqar

#####

```
#####
# Role: ComputeOvsDpdk
#
#####
#####
- name: ComputeOvsDpdk
  description: |
    Compute Ovs DPDK Role
  CountDefault: 1
  networks:
    - InternalApi
    - Tenant
    - Storage
  HostnameFormatDefault: 'computeovsdpdk-%index%'
  disable_upgrade_deployment: True
  ServicesDefault:
    - OS::Triple0::Services::AuditD
    - OS::Triple0::Services::CACerts
    - OS::Triple0::Services::CephClient
    - OS::Triple0::Services::CephExternal
    - OS::Triple0::Services::CertmongerUser
    - OS::Triple0::Services::Collectd
    - OS::Triple0::Services::ComputeCeilometerAgent
    - OS::Triple0::Services::ComputeNeutronCorePlugin
    - OS::Triple0::Services::ComputeNeutronL3Agent
    - OS::Triple0::Services::ComputeNeutronMetadataAgent
    - OS::Triple0::Services::ComputeNeutronOvsDpdk
    - OS::Triple0::Services::Docker
    - OS::Triple0::Services::FluentdClient
    - OS::Triple0::Services::Iscsid
    - OS::Triple0::Services::Kernel
    - OS::Triple0::Services::MySQLClient
    - OS::Triple0::Services::NovaCompute
    - OS::Triple0::Services::NovaLibvirt
    - OS::Triple0::Services::Ntp
    - OS::Triple0::Services::ContainersLogrotateCronD
    - OS::Triple0::Services::OpenDaylightOvs
    - OS::Triple0::Services::Securetty
    - OS::Triple0::Services::SensuClient
    - OS::Triple0::Services::Snmp
    - OS::Triple0::Services::Sshd
    - OS::Triple0::Services::Timezone
    - OS::Triple0::Services::TripleoFirewall
    - OS::Triple0::Services::TripleoPackages
```

B.1.2. post-install.yaml

```
heat_template_version: 2014-10-16

description: >
  Example extra config for post-deployment

parameters:
  servers:
    type: json
```

resources:

ExtraDeployments:

```
type: OS::Heat::StructuredDeployments
properties:
  servers: {get_param: servers}
  config: {get_resource: ExtraConfig}
  actions: ['CREATE', 'UPDATE']
```

ExtraConfig:

```
type: OS::Heat::SoftwareConfig
properties:
  group: script
  config: |
    #!/bin/bash
    set -x
    function tuned_service_dependency() {
      tuned_service=/usr/lib/systemd/system/tuned.service
      grep -q "network.target" $tuned_service
      if [ "$?" -eq 0 ]; then
        sed -i '/After=.*s/network.target//g' $tuned_service
      fi
      grep -q "Before=.*network.target" $tuned_service
      if [ ! "$?" -eq 0 ]; then
        grep -q "Before=.*" $tuned_service
        if [ "$?" -eq 0 ]; then
          sed -i 's/^\(Before=.*\)\/\1 network.target
openvswitch.service/g' $tuned_service
        else
          sed -i '/After/i Before=network.target
openvswitch.service' $tuned_service
        fi
      fi
    }

    if hiera -c /etc/puppet/hiera.yaml service_names | grep -q
neutron_ovs_dpdk_agent; then
      tuned_service_dependency
    fi
```

B.1.3. network.environment.yaml

resource_registry:

Specify the relative/absolute path to the config files you want to use for override the default.

```
OS::TripleO::ComputeOvsDpdk::Net::SoftwareConfig: nic-
configs/computeovsdpdk.yaml
```

```
OS::TripleO::Controller::Net::SoftwareConfig: nic-
configs/controller.yaml
```

```
OS::TripleO::NodeExtraConfigPost: post-install.yaml
```

parameter_defaults:

Customize all these values to match the local environment

```
InternalApiNetCidr: 10.10.10.0/24
```

```

TenantNetCidr: 10.10.2.0/24
StorageNetCidr: 10.10.3.0/24
StorageMgmtNetCidr: 10.10.4.0/24
ExternalNetCidr: 172.20.12.112/28
# CIDR subnet mask length for provisioning network
ControlPlaneSubnetCidr: '24'
InternalApiAllocationPools: [{'start': '10.10.10.10', 'end':
'10.10.10.200'}]
TenantAllocationPools: [{'start': '10.10.2.100', 'end': '10.10.2.200'}]
StorageAllocationPools: [{'start': '10.10.3.100', 'end': '10.10.3.200'}]
StorageMgmtAllocationPools: [{'start': '10.10.4.100', 'end':
'10.10.4.200'}]
# Use an External allocation pool which will leave room for floating IPs
ExternalAllocationPools: [{'start': '172.20.12.114', 'end':
'172.20.12.125'}]
# Set to the router gateway on the external network
ExternalInterfaceDefaultRoute: 172.20.12.126
# Gateway router for the provisioning network (or Undercloud IP)
ControlPlaneDefaultRoute: 192.168.24.1
# Generally the IP of the Undercloud
EC2MetadataIp: 192.168.24.1
InternalApiNetworkVlanID: 10
TenantNetworkVlanID: 11
StorageNetworkVlanID: 12
StorageMgmtNetworkVlanID: 13
ExternalNetworkVlanID: 14
# Define the DNS servers (maximum 2) for the overcloud nodes
DnsServers: ["8.8.8.8", "8.8.4.4"]
# May set to br-ex if using floating IPs only on native VLAN on bridge
br-ex
NeutronExternalNetworkBridge: ""
# The tunnel type for the tenant network (vxlan or gre). Set to '' to
disable tunneling.
NeutronTunnelTypes: ''
# The tenant network type for Neutron (vlan or vxlan).
NeutronNetworkType: 'vlan'
# The OVS logical->physical bridge mappings to use.
NeutronBridgeMappings: 'tenant:br-link0'
# The Neutron ML2 and OpenVSwitch vlan mapping range to support.
NeutronNetworkVLANRanges: 'tenant:22:22,tenant:25:25'
# Nova flavor to use.
OvercloudControllerFlavor: controller
OvercloudComputeOvsDpdkFlavor: compute
#Number of nodes to deploy.
ControllerCount: 1
ComputeOvsDpdkCount: 1
# NTP server configuration.
NtpServer: clock.redhat.com

#####
# OVS DPDK configuration #
# #####
ComputeOvsDpdkParameters:
  KernelArgs: default_hugepagesz=1GB hugepagesz=1G hugepages=32 iommu=pt
intel_iommu=on
  TunedProfileName: "cpu-partitioning"

```

```

    IsolCpusList:
"1,2,3,4,5,6,7,9,10,17,18,19,20,21,22,23,11,12,13,14,15,25,26,27,28,29,30,31"
    NovaVcpuPinSet:
['2,3,4,5,6,7,18,19,20,21,22,23,10,11,12,13,14,15,26,27,28,29,30,31']
    NovaReservedHostMemory: 4096
    OvsDpdkSocketMemory: "1024,1024"
    OvsDpdkMemoryChannels: "4"
    OvsDpdkCoreList: "0,16,8,24"
    OvsPmdCoreList: "1,17,9,25"

# MTU global configuration
NeutronGlobalPhysnetMtu: 9000
# DHCP provide metadata route to VM.
NeutronEnableIsolatedMetadata: true
# DHCP always provides metadata route to VM.
NeutronEnableForceMetadata: true
# Configure the classname of the firewall driver to use for implementing
security groups.
NeutronOVSEthernetDriver: openvswitch

```

B.1.4. controller.yaml

```

heat_template_version: pike

description: >
  Software Config to drive os-net-config to configure VLANs for the
  controller role.

parameters:
  ControlPlaneIp:
    default: ''
    description: IP address/subnet on the ctlplane network
    type: string
  ExternalIpSubnet:
    default: ''
    description: IP address/subnet on the external network
    type: string
  InternalApiIpSubnet:
    default: ''
    description: IP address/subnet on the internal API network
    type: string
  StorageNetworkVlanID:
    default: 30
    description: Vlan ID for the storage network traffic.
    type: number
  StorageMgmtNetworkVlanID:
    default: 40
    description: Vlan ID for the storage mgmt network traffic.
    type: number
  StorageIpSubnet:
    default: ''
    description: IP address/subnet on the storage network
    type: string
  StorageMgmtIpSubnet:

```

```

    default: ''
    description: IP address/subnet on the storage mgmt network
    type: string
  TenantIpSubnet:
    default: ''
    description: IP address/subnet on the tenant network
    type: string
  ManagementIpSubnet: # Only populated when including
environments/network-management.yaml
    default: ''
    description: IP address/subnet on the management network
    type: string
  ExternalNetworkVlanID:
    default: ''
    description: Vlan ID for the external network traffic.
    type: number
  InternalApiNetworkVlanID:
    default: ''
    description: Vlan ID for the internal_api network traffic.
    type: number
  TenantNetworkVlanID:
    default: ''
    description: Vlan ID for the tenant network traffic.
    type: number
  ManagementNetworkVlanID:
    default: 23
    description: Vlan ID for the management network traffic.
    type: number
  ExternalInterfaceDefaultRoute:
    default: ''
    description: default route for the external network
    type: string
  ControlPlaneSubnetCidr: # Override this via parameter_defaults
    default: '24'
    description: The subnet CIDR of the control plane network.
    type: string
  ControlPlaneDefaultRoute: # Override this via parameter_defaults
    description: The default route of the control plane network.
    type: string
  DnsServers: # Override this via parameter_defaults
    default: []
    description: A list of DNS servers (2 max for some implementations)
that will be added to resolv.conf.
    type: comma_delimited_list
  EC2MetadataIp: # Override this via parameter_defaults
    description: The IP address of the EC2 metadata server.
    type: string

resources:
  OsNetConfigImpl:
    type: OS::Heat::SoftwareConfig
    properties:
      group: script
      config:
        str_replace:
          template:

```

```

    get_file: /usr/share/openstack-tripleo-heat-
templates/network/scripts/run-os-net-config.sh
    params:
      $network_config:
        network_config:
          - type: interface
            name: nic1
            use_dhcp: false
            defroute: false

          - type: linux_bond
            name: bond_api
            bonding_options: "mode=active-backup"
            use_dhcp: false
            dns_servers:
              get_param: DnsServers
            addresses:
          - ip_netmask:
              list_join:
                - /
                - - get_param: ControlPlaneIp
                  - get_param: ControlPlaneSubnetCidr
            routes:
          - ip_netmask: 169.254.169.254/32
            next_hop:
              get_param: EC2MetadataIp
            members:
          - type: interface
            name: nic2
            primary: true
          - type: interface
            name: nic3

          - type: vlan
            vlan_id:
              get_param: InternalApiNetworkVlanID
            device: bond_api
            addresses:
          - ip_netmask:
              get_param: InternalApiIpSubnet

          - type: vlan
            vlan_id:
              get_param: TenantNetworkVlanID
            device: bond_api
            addresses:
          - ip_netmask:
              get_param: TenantIpSubnet

          - type: vlan
            vlan_id:
              get_param: StorageNetworkVlanID
            device: bond_api
            addresses:
          - ip_netmask:
              get_param: StorageIpSubnet

```



```

- type: vlan
  vlan_id:
    get_param: StorageMgmtNetworkVlanID
  device: bond_api
  addresses:
    - ip_netmask:
        get_param: StorageMgmtIpSubnet

- type: vlan
  vlan_id:
    get_param: ExternalNetworkVlanID
  device: bond_api
  addresses:
    - ip_netmask:
        get_param: ExternalIpSubnet
  routes:
    - default: true
      next_hop:
        get_param: ExternalInterfaceDefaultRoute

- type: ovs_bridge
  name: br-link0
  use_dhcp: false
  members:
    - type: interface
      name: nic4
      mtu: 9000

outputs:
  OS::stack_id:
    description: The OsNetConfigImpl resource.
    value:
      get_resource: OsNetConfigImpl

```

B.1.5. computeovsdpdk.yaml

```

heat_template_version: pike

description: >
  Software Config to drive os-net-config to configure VLANs for the
  compute role.

parameters:
  ControlPlaneIp:
    default: ''
    description: IP address/subnet on the ctlplane network
    type: string
  ExternalIpSubnet:
    default: ''
    description: IP address/subnet on the external network
    type: string
  InternalApiIpSubnet:
    default: ''
    description: IP address/subnet on the internal API network

```

```

    type: string
TenantIpSubnet:
    default: ''
    description: IP address/subnet on the tenant network
    type: string
ManagementIpSubnet: # Only populated when including
environments/network-management.yaml
    default: ''
    description: IP address/subnet on the management network
    type: string
InternalApiNetworkVlanID:
    default: ''
    description: Vlan ID for the internal_api network traffic.
    type: number
TenantNetworkVlanID:
    default: ''
    description: Vlan ID for the tenant network traffic.
    type: number
ManagementNetworkVlanID:
    default: 23
    description: Vlan ID for the management network traffic.
    type: number
StorageNetworkVlanID:
    default: 30
    description: Vlan ID for the storage network traffic.
    type: number
StorageIpSubnet:
    default: ''
    description: IP address/subnet on the storage network
    type: string
StorageMgmtIpSubnet:
    default: ''
    description: IP address/subnet on the storage mgmt network
    type: string
ControlPlaneSubnetCidr: # Override this via parameter_defaults
    default: '24'
    description: The subnet CIDR of the control plane network.
    type: string
ControlPlaneDefaultRoute: # Override this via parameter_defaults
    description: The default route of the control plane network.
    type: string
DnsServers: # Override this via parameter_defaults
    default: []
    description: A list of DNS servers (2 max for some implementations)
that will be added to resolv.conf.
    type: comma_delimited_list
EC2MetadataIp: # Override this via parameter_defaults
    description: The IP address of the EC2 metadata server.
    type: string
ExternalInterfaceDefaultRoute:
    default: ''
    description: default route for the external network
    type: string

resources:
    OsNetConfigImpl:

```

```

type: OS::Heat::SoftwareConfig
properties:
  group: script
  config:
    str_replace:
      template:
        get_file: /usr/share/openstack-tripleo-heat-
templates/network/scripts/run-os-net-config.sh
    params:
      $network_config:
        network_config:
          - type: interface
            name: nic1
            use_dhcp: false
            defroute: false

          - type: interface
            name: nic2
            use_dhcp: false
            addresses:
              - ip_netmask:
                  list_join:
                    - /
                  - - get_param: ControlPlaneIp
                    - get_param: ControlPlaneSubnetCidr
            routes:
              - ip_netmask: 169.254.169.254/32
                next_hop:
                  get_param: EC2MetadataIp
              - default: true
                next_hop:
                  get_param: ControlPlaneDefaultRoute

          - type: linux_bond
            name: bond_api
            bonding_options: "mode=active-backup"
            use_dhcp: false
            dns_servers:
              get_param: DnsServers
            members:
              - type: interface
                name: nic3
                primary: true
              - type: interface
                name: nic4

          - type: vlan
            vlan_id:
              get_param: InternalApiNetworkVlanID
            device: bond_api
            addresses:
              - ip_netmask:
                  get_param: InternalApiIpSubnet

          - type: vlan
            vlan_id:

```

```

        get_param: TenantNetworkVlanID
    device: bond_api
    addresses:
    - ip_netmask:
        get_param: TenantIpSubnet

- type: vlan
  vlan_id:
    get_param: StorageNetworkVlanID
  device: bond_api
  addresses:
  - ip_netmask:
    get_param: StorageIpSubnet

- type: ovs_user_bridge
  name: br-link0
  use_dhcp: false
  members:
  - type: ovs_dpdk_bond
    name: dpdkbond0
    mtu: 9000
    rx_queue: 2
    members:
    - type: ovs_dpdk_port
      name: dpdk0
      members:
      - type: interface
        name: nic5
    - type: ovs_dpdk_port
      name: dpdk1
      members:
      - type: interface
        name: nic6

```

```

outputs:
  OS::stack_id:
    description: The OsNetConfigImpl resource.
    value:
      get_resource: OsNetConfigImpl

```

B.1.6. overcloud_deploy.sh

```

#!/bin/bash

openstack overcloud deploy \
--templates \
-r /home/stack/ospd-12-vlan-dpdk-two-ports-ctlplane-dataplane-
bonding/roles_data.yaml \
-e /usr/share/openstack-tripleo-heat-templates/environments/network-
isolation.yaml \
-e /usr/share/openstack-tripleo-heat-templates/environments/host-config-
and-reboot.yaml \
-e /usr/share/openstack-tripleo-heat-templates/environments/neutron-ovs-
dpdk.yaml \
-e /home/stack/ospd-12-vlan-dpdk-two-ports-ctlplane-dataplane-

```

```

bonding/docker-images.yaml \
-e /home/stack/ospd-12-vlan-dpdk-two-ports-ctlplane-dataplane-
bonding/network-environment.yaml \
--log-file overcloud_install.log &> overcloud_install.log

```

B.2. SAMPLE VXLAN OVS-DPDK DATA PLANE BONDING YAML FILES

B.2.1. roles_data.yaml

```

#####
#####
# File generated by TripleO
#####
#####
#####
#####
# Role: Controller
#
#####
#####
- name: Controller
  description: |
    Controller role that has all the controller services loaded and handles
    Database, Messaging and Network functions.
  CountDefault: 1
  tags:
    - primary
    - controller
  networks:
    - External
    - InternalApi
    - Storage
    - StorageMgmt
    - Tenant
  HostnameFormatDefault: 'controller-%index%'
  # Deprecated & backward-compatible values (FIXME: Make parameters
consistent)
  # Set uses_deprecated_params to True if any deprecated params are used.
  uses_deprecated_params: True
  deprecated_param_extraconfig: 'controllerExtraConfig'
  deprecated_param_flavor: 'OvercloudControlFlavor'
  deprecated_param_image: 'controllerImage'
  ServicesDefault:
    - OS::TripleO::Services::AodhApi
    - OS::TripleO::Services::AodhEvaluator
    - OS::TripleO::Services::AodhListener
    - OS::TripleO::Services::AodhNotifier
    - OS::TripleO::Services::AuditD
    - OS::TripleO::Services::BarbicanApi
    - OS::TripleO::Services::CACerts
    - OS::TripleO::Services::CeilometerAgentCentral
    - OS::TripleO::Services::CeilometerAgentNotification
    # FIXME: This service was disabled in Pike and this entry should be
removed

```

```
# in Queens.
- OS::Triple0::Services::CeilometerApi
- OS::Triple0::Services::CeilometerCollector
- OS::Triple0::Services::CeilometerExpirer
- OS::Triple0::Services::CephExternal
- OS::Triple0::Services::CephMds
- OS::Triple0::Services::CephMon
- OS::Triple0::Services::CephRbdMirror
- OS::Triple0::Services::CephRgw
- OS::Triple0::Services::CertmongerUser
- OS::Triple0::Services::CinderApi
- OS::Triple0::Services::CinderBackendDellPs
- OS::Triple0::Services::CinderBackendDellSc
- OS::Triple0::Services::CinderBackendDellEMCUnity
- OS::Triple0::Services::CinderBackendDellEMCVMAXISCSI
- OS::Triple0::Services::CinderBackendNetApp
- OS::Triple0::Services::CinderBackendScaleIO
- OS::Triple0::Services::CinderBackendVRTSHyperScale
- OS::Triple0::Services::CinderBackup
- OS::Triple0::Services::CinderHPELeftHandISCSI
- OS::Triple0::Services::CinderScheduler
- OS::Triple0::Services::CinderVolume
- OS::Triple0::Services::Clustercheck
- OS::Triple0::Services::Collectd
- OS::Triple0::Services::Congress
- OS::Triple0::Services::Docker
- OS::Triple0::Services::Ec2Api
- OS::Triple0::Services::Etcd
- OS::Triple0::Services::ExternalSwiftProxy
- OS::Triple0::Services::FluentdClient
- OS::Triple0::Services::GlanceApi
- OS::Triple0::Services::GnocchiApi
- OS::Triple0::Services::GnocchiMetricd
- OS::Triple0::Services::GnocchiStatsd
- OS::Triple0::Services::HAproxy
- OS::Triple0::Services::HeatApi
- OS::Triple0::Services::HeatApiCfn
- OS::Triple0::Services::HeatApiCloudwatch
- OS::Triple0::Services::HeatEngine
- OS::Triple0::Services::Horizon
- OS::Triple0::Services::IronicApi
- OS::Triple0::Services::IronicConductor
- OS::Triple0::Services::Iscsid
- OS::Triple0::Services::Keepalived
- OS::Triple0::Services::Kernel
- OS::Triple0::Services::Keystone
- OS::Triple0::Services::ManilaApi
- OS::Triple0::Services::ManilaBackendCephFs
- OS::Triple0::Services::ManilaBackendGeneric
- OS::Triple0::Services::ManilaBackendIsilon
- OS::Triple0::Services::ManilaBackendNetapp
- OS::Triple0::Services::ManilaBackendUnity
- OS::Triple0::Services::ManilaBackendVNX
- OS::Triple0::Services::ManilaBackendVMAX
- OS::Triple0::Services::ManilaScheduler
- OS::Triple0::Services::ManilaShare
```

- OS::Triple0::Services::Memcached
- OS::Triple0::Services::MongoDb
- OS::Triple0::Services::MySQL
- OS::Triple0::Services::MySQLClient
- OS::Triple0::Services::NeutronApi
- OS::Triple0::Services::NeutronBgpVpnApi
- OS::Triple0::Services::NeutronCorePlugin
- OS::Triple0::Services::NeutronDhcpAgent
- OS::Triple0::Services::NeutronL2gwAgent
- OS::Triple0::Services::NeutronL2gwApi
- OS::Triple0::Services::NeutronL3Agent
- OS::Triple0::Services::NeutronLbaasv2Agent
- OS::Triple0::Services::NeutronLinuxbridgeAgent
- OS::Triple0::Services::NeutronMetadataAgent
- OS::Triple0::Services::NeutronML2FujitsuCfab
- OS::Triple0::Services::NeutronML2FujitsuFossw
- OS::Triple0::Services::NeutronOvsAgent
- OS::Triple0::Services::NeutronVppAgent
- OS::Triple0::Services::NovaApi
- OS::Triple0::Services::NovaConductor
- OS::Triple0::Services::NovaConsoleauth
- OS::Triple0::Services::NovaIronic
- OS::Triple0::Services::NovaMetadata
- OS::Triple0::Services::NovaPlacement
- OS::Triple0::Services::NovaScheduler
- OS::Triple0::Services::NovaVncProxy
- OS::Triple0::Services::Ntp
- OS::Triple0::Services::ContainersLogrotateCron
- OS::Triple0::Services::OctaviaApi
- OS::Triple0::Services::OctaviaHealthManager
- OS::Triple0::Services::OctaviaHousekeeping
- OS::Triple0::Services::OctaviaWorker
- OS::Triple0::Services::OpenDaylightApi
- OS::Triple0::Services::OpenDaylightOvs
- OS::Triple0::Services::OVNDBs
- OS::Triple0::Services::OVNController
- OS::Triple0::Services::Pacemaker
- OS::Triple0::Services::PankoApi
- OS::Triple0::Services::RabbitMQ
- OS::Triple0::Services::Redis
- OS::Triple0::Services::SaharaApi
- OS::Triple0::Services::SaharaEngine
- OS::Triple0::Services::Securetty
- OS::Triple0::Services::SensuClient
- OS::Triple0::Services::Snmp
- OS::Triple0::Services::Sshd
- OS::Triple0::Services::SwiftProxy
- OS::Triple0::Services::SwiftRingBuilder
- OS::Triple0::Services::SwiftStorage
- OS::Triple0::Services::Tacker
- OS::Triple0::Services::Timezone
- OS::Triple0::Services::TripleoFirewall
- OS::Triple0::Services::TripleoPackages
- OS::Triple0::Services::Tuned
- OS::Triple0::Services::Vpp
- OS::Triple0::Services::Zaqar

```
#####
####
# Role: ComputeOvsDpdk
#
#####
####
- name: ComputeOvsDpdk
  description: |
    Compute OvS DPDK Role
  CountDefault: 1
  networks:
    - InternalApi
    - Tenant
    - Storage
  HostnameFormatDefault: 'computeovsdpdk-%index%'
  disable_upgrade_deployment: True
  ServicesDefault:
    - OS::TripleO::Services::AuditD
    - OS::TripleO::Services::CACerts
    - OS::TripleO::Services::CephClient
    - OS::TripleO::Services::CephExternal
    - OS::TripleO::Services::CertmongerUser
    - OS::TripleO::Services::Collectd
    - OS::TripleO::Services::ComputeCeilometerAgent
    - OS::TripleO::Services::ComputeNeutronCorePlugin
    - OS::TripleO::Services::ComputeNeutronL3Agent
    - OS::TripleO::Services::ComputeNeutronMetadataAgent
    - OS::TripleO::Services::ComputeNeutronOvsDpdk
    - OS::TripleO::Services::Docker
    - OS::TripleO::Services::FluentdClient
    - OS::TripleO::Services::Iscsid
    - OS::TripleO::Services::Kernel
    - OS::TripleO::Services::MySQLClient
    - OS::TripleO::Services::NovaCompute
    - OS::TripleO::Services::NovaLibvirt
    - OS::TripleO::Services::Ntp
    - OS::TripleO::Services::ContainersLogrotateCron
    - OS::TripleO::Services::OpenDaylightOvs
    - OS::TripleO::Services::Securetty
    - OS::TripleO::Services::SensuClient
    - OS::TripleO::Services::Snmp
    - OS::TripleO::Services::Sshd
    - OS::TripleO::Services::Timezone
    - OS::TripleO::Services::TripleoFirewall
    - OS::TripleO::Services::TripleoPackages
```

B.2.2. post-install.yaml

```
heat_template_version: 2014-10-16

description: >
  Example extra config for post-deployment

parameters:
  servers:
```



```

    type: json

resources:

  ExtraDeployments:
    type: OS::Heat::StructuredDeployments
    properties:
      servers: {get_param: servers}
      config: {get_resource: ExtraConfig}
      actions: ['CREATE', 'UPDATE']

  ExtraConfig:
    type: OS::Heat::SoftwareConfig
    properties:
      group: script
      config: |
        #!/bin/bash
        set -x
        function tuned_service_dependency() {
          tuned_service=/usr/lib/systemd/system/tuned.service
          grep -q "network.target" $tuned_service
          if [ "$?" -eq 0 ]; then
            sed -i '/After=.*s/network.target//g' $tuned_service
          fi
          grep -q "Before=.*network.target" $tuned_service
          if [ ! "$?" -eq 0 ]; then
            grep -q "Before=.*" $tuned_service
            if [ "$?" -eq 0 ]; then
              sed -i 's/^\(Before=.*\)\/\1 network.target
openvswitch.service/g' $tuned_service
            else
              sed -i '/After/i Before=network.target
openvswitch.service' $tuned_service
            fi
          fi
        }

        if hiera -c /etc/puppet/hiera.yaml service_names | grep -q
neutron_ovs_dpdk_agent; then
          tuned_service_dependency
        fi

```

B.2.3. network.environment.yaml

```

resource_registry:
  # Specify the relative/absolute path to the config files you want to use
  # for override the default.
  OS::Triple0::ComputeOvsDpdk::Net::SoftwareConfig: nic-configs/compute-
  ovs-dpdk.yaml
  OS::Triple0::Controller::Net::SoftwareConfig: nic-
  configs/controller.yaml
  OS::Triple0::NodeExtraConfigPost: post-install.yaml

parameter_defaults:
  # Customize all these values to match the local environment

```

```

InternalApiNetCidr: 10.10.10.0/24
TenantNetCidr: 10.10.2.0/24
StorageNetCidr: 10.10.3.0/24
StorageMgmtNetCidr: 10.10.4.0/24
ExternalNetCidr: 172.20.12.112/28
# CIDR subnet mask length for provisioning network
ControlPlaneSubnetCidr: '24'
InternalApiAllocationPools: [{'start': '10.10.10.10', 'end':
'10.10.10.200'}]
TenantAllocationPools: [{'start': '10.10.2.100', 'end': '10.10.2.200'}]
StorageAllocationPools: [{'start': '10.10.3.100', 'end': '10.10.3.200'}]
StorageMgmtAllocationPools: [{'start': '10.10.4.100', 'end':
'10.10.4.200'}]
# Use an External allocation pool which will leave room for floating IPs
ExternalAllocationPools: [{'start': '172.20.12.114', 'end':
'172.20.12.125'}]
# Set to the router gateway on the external network
ExternalInterfaceDefaultRoute: 172.20.12.126
# Gateway router for the provisioning network (or Undercloud IP)
ControlPlaneDefaultRoute: 192.168.24.1
# Generally the IP of the Undercloud
EC2MetadataIp: 192.168.24.1
InternalApiNetworkVlanID: 10
TenantNetworkVlanID: 11
StorageNetworkVlanID: 12
StorageMgmtNetworkVlanID: 13
ExternalNetworkVlanID: 14
# Define the DNS servers (maximum 2) for the overcloud nodes
DnsServers: ["8.8.8.8", "8.8.4.4"]
# May set to br-ex if using floating IPs only on native VLAN on bridge
br-ex
NeutronExternalNetworkBridge: ""
# The tunnel type for the tenant network (vxlan or gre). Set to '' to
disable tunneling.
NeutronTunnelTypes: 'vxlan'
# The tenant network type for Neutron (vlan or vxlan).
NeutronNetworkType: 'vxlan'
# Nova flavor to use.
OvercloudControllerFlavor: controller
OvercloudComputeOvsDpdkFlavor: compute
#Number of nodes to deploy.
ControllerCount: 1
ComputeOvsDpdkCount: 1
# NTP server configuration.
NtpServer: clock.redhat.com

# DPDK compute node.
ComputeOvsDpdkParameters:
    KernelArgs: default_hugepagesz=1GB hugepagesz=1G hugepages=32 iommu=pt
intel_iommu=on
    TunedProfileName: "cpu-partitioning"
    IsolCpusList:
"1,2,3,4,5,6,7,9,10,17,18,19,20,21,22,23,11,12,13,14,15,25,26,27,28,29,30,
31"
    NovaVcpuPinSet:
['2,3,4,5,6,7,18,19,20,21,22,23,10,11,12,13,14,15,26,27,28,29,30,31']

```

```

NovaReservedHostMemory: 4096
OvsDpdkSocketMemory: "1024,1024"
OvsDpdkMemoryChannels: "4"
OvsDpdkCoreList: "0,16,8,24"
OvsPmdCoreList: "1,17,9,25"

# MTU global configuration
NeutronGlobalPhysnetMtu: 9000
# DHCP provide metadata route to VM.
NeutronEnableIsolatedMetadata: true
# DHCP always provides metadata route to VM.
NeutronEnableForceMetadata: true
# Configure the classname of the firewall driver to use for implementing
security groups.
NeutronOVSFirewallDriver: openvswitch

```

B.2.4. controller.yaml

```

heat_template_version: pike

description: >
  Software Config to drive os-net-config to configure VLANs for the
  controller role.

parameters:
  ControlPlaneIp:
    default: ''
    description: IP address/subnet on the ctlplane network
    type: string
  ExternalIpSubnet:
    default: ''
    description: IP address/subnet on the external network
    type: string
  InternalApiIpSubnet:
    default: ''
    description: IP address/subnet on the internal API network
    type: string
  StorageIpSubnet:
    default: ''
    description: IP address/subnet on the storage network
    type: string
  StorageMgmtIpSubnet:
    default: ''
    description: IP address/subnet on the storage mgmt network
    type: string
  TenantIpSubnet:
    default: ''
    description: IP address/subnet on the tenant network
    type: string
  ManagementIpSubnet: # Only populated when including
environments/network-management.yaml
    default: ''
    description: IP address/subnet on the management network
    type: string
  ExternalNetworkVlanID:

```

```

    default: ''
    description: Vlan ID for the external network traffic.
    type: number
InternalApiNetworkVlanID:
    default: ''
    description: Vlan ID for the internal_api network traffic.
    type: number
StorageNetworkVlanID:
    default: 30
    description: Vlan ID for the storage network traffic.
    type: number
StorageMgmtNetworkVlanID:
    default: 40
    description: Vlan ID for the storage mgmt network traffic.
    type: number
TenantNetworkVlanID:
    default: ''
    description: Vlan ID for the tenant network traffic.
    type: number
ManagementNetworkVlanID:
    default: 23
    description: Vlan ID for the management network traffic.
    type: number
ExternalInterfaceDefaultRoute:
    default: ''
    description: default route for the external network
    type: string
ControlPlaneSubnetCidr: # Override this via parameter_defaults
    default: '24'
    description: The subnet CIDR of the control plane network.
    type: string
ControlPlaneDefaultRoute: # Override this via parameter_defaults
    description: The default route of the control plane network.
    type: string
DnsServers: # Override this via parameter_defaults
    default: []
    description: A list of DNS servers (2 max for some implementations)
that will be added to resolv.conf.
    type: comma_delimited_list
EC2MetadataIp: # Override this via parameter_defaults
    description: The IP address of the EC2 metadata server.
    type: string

resources:
  OsNetConfigImpl:
    type: OS::Heat::SoftwareConfig
    properties:
      group: script
      config:
        str_replace:
          template:
            get_file: /usr/share/openstack-tripleo-heat-
templates/network/scripts/run-os-net-config.sh
        params:
          $network_config:
            network_config:

```

```

- type: interface
  name: nic1
  use_dhcp: false
  defroute: false

- type: linux_bond
  name: bond_api
  bonding_options: "mode=active-backup"
  use_dhcp: false
  dns_servers:
    get_param: DnsServers
  addresses:
    - ip_netmask:
        list_join:
          - /
          - - get_param: ControlPlaneIp
            - get_param: ControlPlaneSubnetCidr
  routes:
    - ip_netmask: 169.254.169.254/32
      next_hop:
        get_param: EC2MetadataIp
  members:
    - type: interface
      name: nic2
      primary: true
    - type: interface
      name: nic3

- type: vlan
  vlan_id:
    get_param: InternalApiNetworkVlanID
  device: bond_api
  addresses:
    - ip_netmask:
        get_param: InternalApiIpSubnet

- type: vlan
  vlan_id:
    get_param: StorageNetworkVlanID
  device: bond_api
  addresses:
    - ip_netmask:
        get_param: StorageIpSubnet

- type: vlan
  vlan_id:
    get_param: StorageMgmtNetworkVlanID
  device: bond_api
  addresses:
    - ip_netmask:
        get_param: StorageMgmtIpSubnet

- type: vlan
  vlan_id:
    get_param: ExternalNetworkVlanID
  device: bond_api

```

```

        addresses:
        - ip_netmask:
            get_param: ExternalIpSubnet
        routes:
        - default: true
            next_hop:
                get_param: ExternalInterfaceDefaultRoute

    - type: ovs_bridge
      name: br-link0
      use_dhcp: false
      members:
        - type: interface
          name: nic4
          mtu: 9000
        - type: vlan
          vlan_id:
            get_param: TenantNetworkVlanID
          addresses:
            - ip_netmask:
                get_param: TenantIpSubnet

  outputs:
    OS::stack_id:
      description: The OsNetConfigImpl resource.
      value:
        get_resource: OsNetConfigImpl

```

B.2.5. compute-ovs-dpdk.yaml

```

heat_template_version: pike

description: >
  Software Config to drive os-net-config to configure VLANs for the
  compute role.

parameters:
  ControlPlaneIp:
    default: ''
    description: IP address/subnet on the ctlplane network
    type: string
  ExternalIpSubnet:
    default: ''
    description: IP address/subnet on the external network
    type: string
  InternalApiIpSubnet:
    default: ''
    description: IP address/subnet on the internal API network
    type: string
  TenantIpSubnet:
    default: ''
    description: IP address/subnet on the tenant network
    type: string
  ManagementIpSubnet: # Only populated when including
    environments/network-management.yaml

```

```

    default: ''
    description: IP address/subnet on the management network
    type: string
InternalApiNetworkVlanID:
    default: ''
    description: Vlan ID for the internal_api network traffic.
    type: number
StorageNetworkVlanID:
    default: 30
    description: Vlan ID for the storage network traffic.
    type: number
StorageMgmtNetworkVlanID:
    default: 40
    description: Vlan ID for the storage mgmt network traffic.
    type: number
TenantNetworkVlanID:
    default: ''
    description: Vlan ID for the tenant network traffic.
    type: number
ManagementNetworkVlanID:
    default: 23
    description: Vlan ID for the management network traffic.
    type: number
StorageIpSubnet:
    default: ''
    description: IP address/subnet on the storage network
    type: string
StorageMgmtIpSubnet:
    default: ''
    description: IP address/subnet on the storage mgmt network
    type: string
ControlPlaneSubnetCidr: # Override this via parameter_defaults
    default: '24'
    description: The subnet CIDR of the control plane network.
    type: string
ControlPlaneDefaultRoute: # Override this via parameter_defaults
    description: The default route of the control plane network.
    type: string
DnsServers: # Override this via parameter_defaults
    default: []
    description: A list of DNS servers (2 max for some implementations)
that will be added to resolv.conf.
    type: comma_delimited_list
EC2MetadataIp: # Override this via parameter_defaults
    description: The IP address of the EC2 metadata server.
    type: string
ExternalInterfaceDefaultRoute:
    default: ''
    description: default route for the external network
    type: string

resources:
  OsNetConfigImpl:
    type: OS::Heat::SoftwareConfig
    properties:
      group: script

```

```

config:
  str_replace:
    template:
      get_file: /usr/share/openstack-tripleo-heat-
templates/network/scripts/run-os-net-config.sh
    params:
      $network_config:
        network_config:
          - type: interface
            name: nic1
            use_dhcp: false
            defroute: false

          - type: interface
            name: nic2
            use_dhcp: false
            addresses:
              - ip_netmask:
                  list_join:
                    - /
                    - - get_param: ControlPlaneIp
                      - get_param: ControlPlaneSubnetCidr
            routes:
              - ip_netmask: 169.254.169.254/32
                next_hop:
                  get_param: EC2MetadataIp
              - default: true
                next_hop:
                  get_param: ControlPlaneDefaultRoute

          - type: linux_bond
            name: bond_api
            bonding_options: "mode=active-backup"
            use_dhcp: false
            dns_servers:
              get_param: DnsServers
            members:
              - type: interface
                name: nic3
                primary: true
              - type: interface
                name: nic4

          - type: vlan
            vlan_id:
              get_param: InternalApiNetworkVlanID
            device: bond_api
            addresses:
              - ip_netmask:
                  get_param: InternalApiIpSubnet

          - type: vlan
            vlan_id:
              get_param: StorageNetworkVlanID
            device: bond_api
            addresses:

```



```

      - ip_netmask:
        get_param: StorageIpSubnet

- type: ovs_user_bridge
  name: br-link0
  use_dhcp: false
  ovs_extra:
    - str_replace:
        template: set port br-link0 tag=_VLAN_TAG_
        params:
          _VLAN_TAG_:
            get_param: TenantNetworkVlanID
  addresses:
    - ip_netmask:
        get_param: TenantIpSubnet
  members:
    - type: ovs_dpdk_bond
      name: dpdkbond0
      mtu: 9000
      rx_queue: 2
      members:
        - type: ovs_dpdk_port
          name: dpdk0
          members:
            - type: interface
              name: nic5
        - type: ovs_dpdk_port
          name: dpdk1
          members:
            - type: interface
              name: nic6

outputs:
  OS::stack_id:
    description: The OsNetConfigImpl resource.
    value:
      get_resource: OsNetConfigImpl

```

B.2.6. overcloud_deploy.sh

```

#!/bin/bash

openstack overcloud deploy \
--templates \
-r /home/stack/ospd-12-vxlan-dpdk-single-port-ctlplane-
bonding/roles_data.yaml \
-e /usr/share/openstack-tripleo-heat-templates/environments/network-
isolation.yaml \
-e /usr/share/openstack-tripleo-heat-templates/environments/host-config-
and-reboot.yaml \
-e /usr/share/openstack-tripleo-heat-templates/environments/neutron-ovs-
dpdk.yaml \
-e /home/stack/ospd-12-vxlan-dpdk-single-port-ctlplane-bonding/docker-
images.yaml \

```

```
-e /home/stack/ospd-12-vxlan-dpdk-single-port-ctlplane-bonding/network-  
environment.yaml \  
--log-file overcloud_install.log &> overcloud_install.log
```

APPENDIX C. SAMPLE HETEROGENEOUS CLUSTER YAML FILES

This section provides sample YAML files as a reference for compute nodes in a heterogeneous cluster.

C.1. SAMPLE SRIOV AND DPDK IN A HETEROGENEOUS COMPUTE CLUSTER YAML FILES

C.1.1. roles_data.yaml

```
#####
#####
# File generated by TripleO
#####
#####
#####
#####
# Role: Controller
#
#####
#####
- name: Controller
  description: |
    Controller role that has all the controller services loaded and handles
    Database, Messaging and Network functions.
  CountDefault: 1
  tags:
    - primary
    - controller
  networks:
    - External
    - InternalApi
    - Storage
    - StorageMgmt
    - Tenant
  HostnameFormatDefault: 'controller-%index%'
  ServicesDefault:
    - OS::TripleO::Services::AodhApi
    - OS::TripleO::Services::AodhEvaluator
    - OS::TripleO::Services::AodhListener
    - OS::TripleO::Services::AodhNotifier
    - OS::TripleO::Services::AuditD
    - OS::TripleO::Services::BarbicanApi
    - OS::TripleO::Services::CACerts
    - OS::TripleO::Services::CeilometerAgentCentral
    - OS::TripleO::Services::CeilometerAgentNotification
    # FIXME: This service was disabled in Pike and this entry should be
    removed
    # in Queens.
    - OS::TripleO::Services::CeilometerExpirer
    - OS::TripleO::Services::CephExternal
    - OS::TripleO::Services::CephMds
    - OS::TripleO::Services::CephMon
```

- OS::Triple0::Services::CephRbdMirror
- OS::Triple0::Services::CephRgw
- OS::Triple0::Services::CertmongerUser
- OS::Triple0::Services::CinderApi
- OS::Triple0::Services::CinderBackendDellPs
- OS::Triple0::Services::CinderBackendDellSc
- OS::Triple0::Services::CinderBackendNetApp
- OS::Triple0::Services::CinderBackendScaleIO
- OS::Triple0::Services::CinderBackendVRTSHyperScale
- OS::Triple0::Services::CinderBackup
- OS::Triple0::Services::CinderHPELeftHandISCSI
- OS::Triple0::Services::CinderScheduler
- OS::Triple0::Services::CinderVolume
- OS::Triple0::Services::Clustercheck
- OS::Triple0::Services::Collectd
- OS::Triple0::Services::Congress
- OS::Triple0::Services::Docker
- OS::Triple0::Services::Ec2Api
- OS::Triple0::Services::Etcd
- OS::Triple0::Services::ExternalSwiftProxy
- OS::Triple0::Services::FluentdClient
- OS::Triple0::Services::GlanceApi
- OS::Triple0::Services::GnocchiApi
- OS::Triple0::Services::GnocchiMetricd
- OS::Triple0::Services::GnocchiStatsd
- OS::Triple0::Services::HAproxy
- OS::Triple0::Services::HeatApi
- OS::Triple0::Services::HeatApiCfn
- OS::Triple0::Services::HeatApiCloudwatch
- OS::Triple0::Services::HeatEngine
- OS::Triple0::Services::Horizon
- OS::Triple0::Services::IronicApi
- OS::Triple0::Services::IronicConductor
- OS::Triple0::Services::Iscsid
- OS::Triple0::Services::Keepalived
- OS::Triple0::Services::Kernel
- OS::Triple0::Services::Keystone
- OS::Triple0::Services::ManilaApi
- OS::Triple0::Services::ManilaBackendCephFs
- OS::Triple0::Services::ManilaBackendGeneric
- OS::Triple0::Services::ManilaBackendNetapp
- OS::Triple0::Services::ManilaScheduler
- OS::Triple0::Services::ManilaShare
- OS::Triple0::Services::Memcached
- OS::Triple0::Services::MongoDb
- OS::Triple0::Services::MySQL
- OS::Triple0::Services::MySQLClient
- OS::Triple0::Services::NeutronApi
- OS::Triple0::Services::NeutronBgpVpnApi
- OS::Triple0::Services::NeutronCorePlugin
- OS::Triple0::Services::NeutronDhcpAgent
- OS::Triple0::Services::NeutronL2gwAgent
- OS::Triple0::Services::NeutronL2gwApi
- OS::Triple0::Services::NeutronL3Agent
- OS::Triple0::Services::NeutronLbaasv2Agent
- OS::Triple0::Services::NeutronLinuxbridgeAgent

```

- OS::Triple0::Services::NeutronMetadataAgent
- OS::Triple0::Services::NeutronML2FujitsuCfab
- OS::Triple0::Services::NeutronML2FujitsuFossw
- OS::Triple0::Services::NeutronOvsAgent
- OS::Triple0::Services::NeutronVppAgent
- OS::Triple0::Services::NovaApi
- OS::Triple0::Services::NovaConductor
- OS::Triple0::Services::NovaConsoleauth
- OS::Triple0::Services::NovaIronic
- OS::Triple0::Services::NovaMetadata
- OS::Triple0::Services::NovaPlacement
- OS::Triple0::Services::NovaScheduler
- OS::Triple0::Services::NovaVncProxy
- OS::Triple0::Services::Ntp
- OS::Triple0::Services::OctaviaApi
- OS::Triple0::Services::OctaviaHealthManager
- OS::Triple0::Services::OctaviaHousekeeping
- OS::Triple0::Services::OctaviaWorker
- OS::Triple0::Services::OpenDaylightApi
- OS::Triple0::Services::OpenDaylightOvs
- OS::Triple0::Services::OVNDBs
- OS::Triple0::Services::OVNController
- OS::Triple0::Services::Pacemaker
- OS::Triple0::Services::PankoApi
- OS::Triple0::Services::RabbitMQ
- OS::Triple0::Services::Redis
- OS::Triple0::Services::SaharaApi
- OS::Triple0::Services::SaharaEngine
- OS::Triple0::Services::Securetty
- OS::Triple0::Services::SensuClient
- OS::Triple0::Services::Snmp
- OS::Triple0::Services::Sshd
- OS::Triple0::Services::SwiftProxy
- OS::Triple0::Services::SwiftRingBuilder
- OS::Triple0::Services::SwiftStorage
- OS::Triple0::Services::Tacker
- OS::Triple0::Services::Timezone
- OS::Triple0::Services::TripleoFirewall
- OS::Triple0::Services::TripleoPackages
- OS::Triple0::Services::Tuned
- OS::Triple0::Services::Vpp
- OS::Triple0::Services::Zaqar

```

```
#####
```

```
####
```

```
# Role: ComputeSriov
```

```
#
```

```
#####
```

```
####
```

```

- name: ComputeSriov
  description: |
    Compute SR-IOV role
  CountDefault: 1
  networks:
    - InternalApi
    - Tenant
    - Storage

```

```

HostnameFormatDefault: 'computesriov-%index%'
disable_upgrade_deployment: True
ServicesDefault:
  - OS::Triple0::Services::AuditD
  - OS::Triple0::Services::CACerts
  - OS::Triple0::Services::CephClient
  - OS::Triple0::Services::CephExternal
  - OS::Triple0::Services::CertmongerUser
  - OS::Triple0::Services::Collectd
  - OS::Triple0::Services::ComputeCeilometerAgent
  - OS::Triple0::Services::ComputeNeutronCorePlugin
  - OS::Triple0::Services::ComputeNeutronL3Agent
  - OS::Triple0::Services::ComputeNeutronMetadataAgent
  - OS::Triple0::Services::ComputeNeutronOvsAgent
  - OS::Triple0::Services::Docker
  - OS::Triple0::Services::FluentdClient
  - OS::Triple0::Services::Iscsid
  - OS::Triple0::Services::Kernel
  - OS::Triple0::Services::MySQLClient
  - OS::Triple0::Services::NeutronLinuxbridgeAgent
  - OS::Triple0::Services::NeutronSriovAgent
  - OS::Triple0::Services::NeutronVppAgent
  - OS::Triple0::Services::NovaCompute
  - OS::Triple0::Services::NovaLibvirt
  - OS::Triple0::Services::NovaMigrationTarget
  - OS::Triple0::Services::Ntp
  - OS::Triple0::Services::OpenDaylightOvs
  - OS::Triple0::Services::Securetty
  - OS::Triple0::Services::SensuClient
  - OS::Triple0::Services::Snmp
  - OS::Triple0::Services::Sshd
  - OS::Triple0::Services::Timezone
  - OS::Triple0::Services::TripleoFirewall
  - OS::Triple0::Services::TripleoPackages
  - OS::Triple0::Services::Tuned
  - OS::Triple0::Services::Vpp
  - OS::Triple0::Services::OVNController
#####
#####
# Role: ComputeOvsDpdk
#
#####
#####
- name: ComputeOvsDpdk
  description: |
    Compute OVS DPDK Role
  CountDefault: 1
  networks:
    - InternalApi
    - Tenant
    - Storage
  HostnameFormatDefault: 'computeovsdpdk-%index%'
  disable_upgrade_deployment: True
  ServicesDefault:
    - OS::Triple0::Services::AuditD
    - OS::Triple0::Services::CACerts

```

```

- OS::Triple0::Services::CephClient
- OS::Triple0::Services::CephExternal
- OS::Triple0::Services::CertmongerUser
- OS::Triple0::Services::Collectd
- OS::Triple0::Services::ComputeCeilometerAgent
- OS::Triple0::Services::ComputeNeutronCorePlugin
- OS::Triple0::Services::ComputeNeutronL3Agent
- OS::Triple0::Services::ComputeNeutronMetadataAgent
- OS::Triple0::Services::ComputeNeutronOvsDpdk
- OS::Triple0::Services::Docker
- OS::Triple0::Services::FluentdClient
- OS::Triple0::Services::Iscsid
- OS::Triple0::Services::Kernel
- OS::Triple0::Services::MySQLClient
- OS::Triple0::Services::NovaCompute
- OS::Triple0::Services::NovaLibvirt
- OS::Triple0::Services::Ntp
- OS::Triple0::Services::OpenDaylightOvs
- OS::Triple0::Services::Securetty
- OS::Triple0::Services::SensuClient
- OS::Triple0::Services::Snmp
- OS::Triple0::Services::Sshd
- OS::Triple0::Services::Timezone
- OS::Triple0::Services::TripleoFirewall
- OS::Triple0::Services::TripleoPackages

```

C.1.2. post-install.yaml

```

heat_template_version: 2014-10-16

description: >
  Example extra config for post-deployment

parameters:
  servers:
    type: json

resources:

  ExtraDeployments:
    type: OS::Heat::StructuredDeployments
    properties:
      servers: {get_param: servers}
      config: {get_resource: ExtraConfig}
      actions: ['CREATE', 'UPDATE']

  ExtraConfig:
    type: OS::Heat::SoftwareConfig
    properties:
      group: script
      config: |
        #!/bin/bash
        set -x
        function tuned_service_dependency() {
          tuned_service=/usr/lib/systemd/system/tuned.service

```

```

        grep -q "network.target" $tuned_service
        if [ "$?" -eq 0 ]; then
            sed -i '/After=.*s/network.target//g' $tuned_service
        fi
        grep -q "Before=.*network.target" $tuned_service
        if [ ! "$?" -eq 0 ]; then
            grep -q "Before=.*" $tuned_service
            if [ "$?" -eq 0 ]; then
                sed -i 's/^\(Before=.*\)\/\1 network.target
openvswitch.service/g' $tuned_service
            else
                sed -i '/After/i Before=network.target
openvswitch.service' $tuned_service
            fi
        fi
    }

    if hiera -c /etc/puppet/hiera.yaml service_names | grep -q
neutron_ovs_dpdk_agent; then
        tuned_service_dependency
    fi

```

C.1.3. network.environment.yaml

```

resource_registry:
    # Specify the relative/absolute path to the config files you want to use
    # for override the default.
    OS::Triple0::ComputeSriov::Net::SoftwareConfig: nic-configs/compute-
sriov.yaml
    OS::Triple0::ComputeOvsDpdk::Net::SoftwareConfig: nic-configs/compute-
ovs-dpdk.yaml
    OS::Triple0::Controller::Net::SoftwareConfig: nic-
configs/controller.yaml
    OS::Triple0::NodeExtraConfigPost: post-install.yaml

parameter_defaults:
    # MTU global configuration
    NeutronGlobalPhysnetMtu: 9000
    # Customize all these values to match the local environment
    InternalApiNetCidr: 10.10.10.0/24
    TenantNetCidr: 10.10.2.0/24
    StorageNetCidr: 10.10.3.0/24
    StorageMgmtNetCidr: 10.10.4.0/24
    ExternalNetCidr: 172.20.12.112/28
    # CIDR subnet mask length for provisioning network
    ControlPlaneSubnetCidr: '24'
    InternalApiAllocationPools: [{'start': '10.10.10.10', 'end':
'10.10.10.200'}]
    TenantAllocationPools: [{'start': '10.10.2.100', 'end': '10.10.2.200'}]
    StorageAllocationPools: [{'start': '10.10.3.100', 'end': '10.10.3.200'}]
    StorageMgmtAllocationPools: [{'start': '10.10.4.100', 'end':
'10.10.4.200'}]
    # Use an External allocation pool which will leave room for floating IPs
    ExternalAllocationPools: [{'start': '172.20.12.114', 'end':
'172.20.12.125'}]

```



```

# Set to the router gateway on the external network
ExternalInterfaceDefaultRoute: 172.20.12.126
# Gateway router for the provisioning network (or Undercloud IP)
ControlPlaneDefaultRoute: 192.168.24.1
# Generally the IP of the Undercloud
EC2MetadataIp: 192.168.24.1
InternalApiNetworkVlanID: 10
TenantNetworkVlanID: 11
StorageNetworkVlanID: 12
StorageMgmtNetworkVlanID: 13
ExternalNetworkVlanID: 14
# Define the DNS servers (maximum 2) for the overcloud nodes
DnsServers: ["8.8.8.8", "8.8.4.4"]
# May set to br-ex if using floating IPs only on native VLAN on bridge
br-ex
NeutronExternalNetworkBridge: ""
# The tunnel type for the tenant network (vxlan or gre). Set to '' to
disable tunneling.
NeutronTunnelTypes: ''
# The tenant network type for Neutron (vlan or vxlan).
NeutronNetworkType: 'vlan'
# The OVS logical->physical bridge mappings to use.
NeutronBridgeMappings: 'tenant:br-link0'
# The Neutron ML2 and OpenVSwitch vlan mapping range to support.
NeutronNetworkVLANRanges: 'tenant:22:22,tenant:25:25'
# Nova flavor to use.
OvercloudControlFlavor: controller
OvercloudComputeOvsDpdkFlavor: computeovsdpdk
OvercloudComputeSriovFlavor: computesriov
#Number of nodes to deploy.
ControllerCount: 1
ComputeOvsDpdkCount: 1
ComputeSriovCount: 1
# NTP server configuration.
NtpServer: clock.redhat.com

#####
# SRIOV configuration #
#####
# The mechanism drivers for the Neutron tenant network.
NeutronMechanismDrivers: "openvswitch,sriovnicswitch"
# List of PCI Passthrough whitelist parameters.
# Use ONE of the following examples.
# Example 1:
# NovaPCIPassthrough:
#   - vendor_id: "8086"
#     product_id: "154c"
#     address: "0000:05:00.0" - (optional)
#     physical_network: "datacentre"
#
# Example 2:
# NovaPCIPassthrough:
#   - devname: "p6p1"
#     physical_network: "tenant"
NovaPCIPassthrough:
  - devname: "ens2f0"

```

```

    physical_network: "tenant"
    - devname: "ens2f1"
    physical_network: "tenant"
# List of supported pci vendor devices in the format VendorID:ProductID.
NeutronSupportedPCIVendorDevs: ['8086:154d', '8086:10ed']
# List of <physical_network>:<physical device>
# All physical networks listed in network_vlan_ranges on the server
# should have mappings to appropriate interfaces on each agent.
NeutronPhysicalDevMappings: "tenant:ens2f0,tenant:ens2f1"
# Provide the list of VFs to be reserved for each SR-IOV interface.
# Format "<interface_name1>:<numvfs1>", "<interface_name2>:<numvfs2>"
# Example "eth1:4096", "eth2:128"
NeutronSriovNumVFs: "ens2f0:5,ens2f1:5"

#####
# Additional computes configuration #
#####

# SR-IOV compute node.
ComputeSriovParameters:
    KernelArgs: default_hugepagesz=1GB hugepagesz=1G hugepages=32 iommu=pt
intel_iommu=on
    TunedProfileName: "cpu-partitioning"
    IsolCpusList:
"1,2,3,4,5,6,7,9,10,17,18,19,20,21,22,23,11,12,13,14,15,25,26,27,28,29,30,
31"
    NovaVcpuPinSet:
['2,3,4,5,6,7,18,19,20,21,22,23,10,11,12,13,14,15,26,27,28,29,30,31']
    NovaReservedHostMemory: 4096

# DPDK compute node.
ComputeOvsDpdkParameters:
    KernelArgs: default_hugepagesz=1GB hugepagesz=1G hugepages=32 iommu=pt
intel_iommu=on
    TunedProfileName: "cpu-partitioning"
    IsolCpusList:
"1,2,3,4,5,6,7,9,10,17,18,19,20,21,22,23,11,12,13,14,15,25,26,27,28,29,30,
31"
    NovaVcpuPinSet:
['2,3,4,5,6,7,18,19,20,21,22,23,10,11,12,13,14,15,26,27,28,29,30,31']
    NovaReservedHostMemory: 4096
    OvsDpdkSocketMemory: "1024,1024"
    OvsDpdkMemoryChannels: "4"
    OvsDpdkCoreList: "0,16,8,24"
    OvsPmdCoreList: "1,17,9,25"

# DHCP provide metadata route to VM.
NeutronEnableIsolatedMetadata: true
# DHCP always provides metadata route to VM.
NeutronEnableForceMetadata: true
# Configure the classname of the firewall driver to use for implementing
security groups.
NeutronOVSVFirewallDriver: openvswitch

# List of scheduler available filters
NovaSchedulerAvailableFilters:

```

```
[ "nova.scheduler.filters.all_filters", "nova.scheduler.filters.pci_passthrough_filter.PciPassthroughFilter" ]
# An array of filters used by Nova to filter a node. These filters will
# be applied in the order they are listed,
# so place your most restrictive filters first to make the filtering
# process more efficient.
NovaSchedulerDefaultFilters:
[ 'AvailabilityZoneFilter', 'RamFilter', 'ComputeFilter', 'ComputeCapabilitiesFilter', 'ImagePropertiesFilter', 'ServerGroupAntiAffinityFilter', 'ServerGroupAffinityFilter', 'PciPassthroughFilter' ]
```

C.1.4. controller.yaml

```
heat_template_version: pike

description: >
  Software Config to drive os-net-config to configure VLANs for the
  controller role.

parameters:
  ControlPlaneIp:
    default: ''
    description: IP address/subnet on the ctlplane network
    type: string
  ExternalIpSubnet:
    default: ''
    description: IP address/subnet on the external network
    type: string
  InternalApiIpSubnet:
    default: ''
    description: IP address/subnet on the internal API network
    type: string
  StorageIpSubnet:
    default: ''
    description: IP address/subnet on the storage network
    type: string
  StorageMgmtIpSubnet:
    default: ''
    description: IP address/subnet on the storage mgmt network
    type: string
  TenantIpSubnet:
    default: ''
    description: IP address/subnet on the tenant network
    type: string
  ManagementIpSubnet: # Only populated when including
environments/network-management.yaml
    default: ''
    description: IP address/subnet on the management network
    type: string
  ExternalNetworkVlanID:
    default: ''
    description: Vlan ID for the external network traffic.
    type: number
  InternalApiNetworkVlanID:
    default: ''
```

```

    description: Vlan ID for the internal_api network traffic.
    type: number
StorageNetworkVlanID:
    default: 30
    description: Vlan ID for the storage network traffic.
    type: number
StorageMgmtNetworkVlanID:
    default: 40
    description: Vlan ID for the storage mgmt network traffic.
    type: number
TenantNetworkVlanID:
    default: ''
    description: Vlan ID for the tenant network traffic.
    type: number
ManagementNetworkVlanID:
    default: 23
    description: Vlan ID for the management network traffic.
    type: number
ExternalInterfaceDefaultRoute:
    default: ''
    description: default route for the external network
    type: string
ControlPlaneSubnetCidr: # Override this via parameter_defaults
    default: '24'
    description: The subnet CIDR of the control plane network.
    type: string
ControlPlaneDefaultRoute: # Override this via parameter_defaults
    description: The default route of the control plane network.
    type: string
DnsServers: # Override this via parameter_defaults
    default: []
    description: A list of DNS servers (2 max for some implementations)
that will be added to resolv.conf.
    type: comma_delimited_list
EC2MetadataIp: # Override this via parameter_defaults
    description: The IP address of the EC2 metadata server.
    type: string

resources:
  OsNetConfigImpl:
    type: OS::Heat::SoftwareConfig
    properties:
      group: script
      config:
        str_replace:
          template:
            get_file: /usr/share/openstack-tripleo-heat-
templates/network/scripts/run-os-net-config.sh
        params:
          $network_config:
            network_config:
              - type: interface
                name: nic1
                use_dhcp: false
                defroute: false

```

```

- type: linux_bond
  name: bond_api
  bonding_options: "mode=active-backup"
  use_dhcp: false
  dns_servers:
    get_param: DnsServers
  addresses:
  - ip_netmask:
      list_join:
        - /
        - - get_param: ControlPlaneIp
          - get_param: ControlPlaneSubnetCidr
  routes:
  - ip_netmask: 169.254.169.254/32
    next_hop:
      get_param: EC2MetadataIp
  members:
  - type: interface
    name: nic2
    primary: true
  - type: interface
    name: nic3

- type: vlan
  vlan_id:
    get_param: InternalApiNetworkVlanID
  device: bond_api
  addresses:
  - ip_netmask:
      get_param: InternalApiIpSubnet

- type: vlan
  vlan_id:
    get_param: TenantNetworkVlanID
  device: bond_api
  addresses:
  - ip_netmask:
      get_param: TenantIpSubnet

- type: vlan
  vlan_id:
    get_param: StorageNetworkVlanID
  device: bond_api
  addresses:
  - ip_netmask:
      get_param: StorageIpSubnet

- type: vlan
  vlan_id:
    get_param: StorageMgmtNetworkVlanID
  device: bond_api
  addresses:
  - ip_netmask:
      get_param: StorageMgmtIpSubnet

- type: vlan

```

```

        vlan_id:
          get_param: ExternalNetworkVlanID
        device: bond_api
        addresses:
          - ip_netmask:
              get_param: ExternalIpSubnet
        routes:
          - default: true
            next_hop:
              get_param: ExternalInterfaceDefaultRoute

    - type: ovs_bridge
      name: br-link0
      use_dhcp: false
      members:
        - type: interface
          name: nic4
          mtu: 9000

outputs:
  OS::stack_id:
    description: The OsNetConfigImpl resource.
    value:
      get_resource: OsNetConfigImpl

```

C.1.5. compute-ovs-dpdk.yaml

```

heat_template_version: pike

description: >
  Software Config to drive os-net-config to configure VLANs for the
  compute role.

parameters:
  ControlPlaneIp:
    default: ''
    description: IP address/subnet on the ctlplane network
    type: string
  ExternalIpSubnet:
    default: ''
    description: IP address/subnet on the external network
    type: string
  InternalApiIpSubnet:
    default: ''
    description: IP address/subnet on the internal API network
    type: string
  TenantIpSubnet:
    default: ''
    description: IP address/subnet on the tenant network
    type: string
  ManagementIpSubnet: # Only populated when including
environments/network-management.yaml
    default: ''
    description: IP address/subnet on the management network
    type: string

```

```

InternalApiNetworkVlanID:
  default: ''
  description: Vlan ID for the internal_api network traffic.
  type: number
TenantNetworkVlanID:
  default: ''
  description: Vlan ID for the tenant network traffic.
  type: number
ManagementNetworkVlanID:
  default: 23
  description: Vlan ID for the management network traffic.
  type: number
StorageNetworkVlanID:
  default: 30
  description: Vlan ID for the storage network traffic.
  type: number
StorageIpSubnet:
  default: ''
  description: IP address/subnet on the storage network
  type: string
StorageMgmtIpSubnet:
  default: ''
  description: IP address/subnet on the storage mgmt network
  type: string
ControlPlaneSubnetCidr: # Override this via parameter_defaults
  default: '24'
  description: The subnet CIDR of the control plane network.
  type: string
ControlPlaneDefaultRoute: # Override this via parameter_defaults
  description: The default route of the control plane network.
  type: string
DnsServers: # Override this via parameter_defaults
  default: []
  description: A list of DNS servers (2 max for some implementations)
that will be added to resolv.conf.
  type: comma_delimited_list
EC2MetadataIp: # Override this via parameter_defaults
  description: The IP address of the EC2 metadata server.
  type: string
ExternalInterfaceDefaultRoute:
  default: ''
  description: default route for the external network
  type: string

resources:
  OsNetConfigImpl:
    type: OS::Heat::SoftwareConfig
    properties:
      group: script
      config:
        str_replace:
          template:
            get_file: /usr/share/openstack-tripleo-heat-
templates/network/scripts/run-os-net-config.sh
        params:
          $network_config:

```

```

network_config:
- type: interface
  name: nic1
  use_dhcp: false
  defroute: false

- type: interface
  name: nic2
  use_dhcp: false
  addresses:
  - ip_netmask:
      list_join:
      - /
      - - get_param: ControlPlaneIp
        - get_param: ControlPlaneSubnetCidr
  routes:
  - ip_netmask: 169.254.169.254/32
    next_hop:
      get_param: EC2MetadataIp
  - default: true
    next_hop:
      get_param: ControlPlaneDefaultRoute

- type: linux_bond
  name: bond_api
  bonding_options: "mode=active-backup"
  use_dhcp: false
  dns_servers:
    get_param: DnsServers
  members:
  - type: interface
    name: nic3
    primary: true
  - type: interface
    name: nic4

- type: vlan
  vlan_id:
    get_param: InternalApiNetworkVlanID
  device: bond_api
  addresses:
  - ip_netmask:
      get_param: InternalApiIpSubnet

- type: vlan
  vlan_id:
    get_param: TenantNetworkVlanID
  device: bond_api
  addresses:
  - ip_netmask:
      get_param: TenantIpSubnet

- type: vlan
  vlan_id:
    get_param: StorageNetworkVlanID
  device: bond_api

```



```

        addresses:
        - ip_netmask:
            get_param: StorageIpSubnet

        - type: ovs_user_bridge
          name: br-link0
          use_dhcp: false
          members:
            - type: ovs_dpdk_bond
              name: dpdkbond0
              mtu: 9000
              rx_queue: 2
              members:
                - type: ovs_dpdk_port
                  name: dpdk0
                  mtu: 9000
                  members:
                    - type: interface
                      name: nic5
                - type: ovs_dpdk_port
                  name: dpdk1
                  mtu: 9000
                  members:
                    - type: interface
                      name: nic6

outputs:
  OS::stack_id:
    description: The OsNetConfigImpl resource.
    value:
      get_resource: OsNetConfigImpl

```

C.1.6. compute-sriov.yaml

```

heat_template_version: pike

description: >
  Software Config to drive os-net-config to configure VLANs for the
  compute role.

parameters:
  ControlPlaneIp:
    default: ''
    description: IP address/subnet on the ctlplane network
    type: string
  ExternalIpSubnet:
    default: ''
    description: IP address/subnet on the external network
    type: string
  InternalApiIpSubnet:
    default: ''
    description: IP address/subnet on the internal API network
    type: string
  TenantIpSubnet:
    default: ''

```

```

    description: IP address/subnet on the tenant network
    type: string
  ManagementIpSubnet: # Only populated when including
environments/network-management.yaml
    default: ''
    description: IP address/subnet on the management network
    type: string
  InternalApiNetworkVlanID:
    default: ''
    description: Vlan ID for the internal_api network traffic.
    type: number
  StorageNetworkVlanID:
    default: 30
    description: Vlan ID for the storage network traffic.
    type: number
  StorageMgmtNetworkVlanID:
    default: 40
    description: Vlan ID for the storage mgmt network traffic.
    type: number
  TenantNetworkVlanID:
    default: ''
    description: Vlan ID for the tenant network traffic.
    type: number
  ManagementNetworkVlanID:
    default: 23
    description: Vlan ID for the management network traffic.
    type: number
  StorageIpSubnet:
    default: ''
    description: IP address/subnet on the storage network
    type: string
  StorageMgmtIpSubnet:
    default: ''
    description: IP address/subnet on the storage mgmt network
    type: string
  ControlPlaneSubnetCidr: # Override this via parameter_defaults
    default: '24'
    description: The subnet CIDR of the control plane network.
    type: string
  ControlPlaneDefaultRoute: # Override this via parameter_defaults
    description: The default route of the control plane network.
    type: string
  DnsServers: # Override this via parameter_defaults
    default: []
    description: A list of DNS servers (2 max for some implementations)
that will be added to resolv.conf.
    type: comma_delimited_list
  EC2MetadataIp: # Override this via parameter_defaults
    description: The IP address of the EC2 metadata server.
    type: string
  ExternalInterfaceDefaultRoute:
    default: ''
    description: default route for the externalheat stack-list network
    type: string

resources:

```

```

OsNetConfigImpl:
  type: OS::Heat::SoftwareConfig
  properties:
    group: script
    config:
      str_replace:
        template:
          get_file: /usr/share/openstack-tripleo-heat-
templates/network/scripts/run-os-net-config.sh
      params:
        $network_config:
          network_config:
            - type: interface
              name: nic1
              use_dhcp: false
              defroute: false

            - type: interface
              name: nic2
              addresses:
                - ip_netmask:
                    list_join:
                      - /
                      - - get_param: ControlPlaneIp
                        - get_param: ControlPlaneSubnetCidr
              routes:
                - ip_netmask: 169.254.169.254/32
                  next_hop:
                    get_param: EC2MetadataIp
                - next_hop:
                    get_param: ControlPlaneDefaultRoute

            - type: linux_bond
              name: bond_api
              bonding_options: "mode=active-backup"
              use_dhcp: false
              dns_servers:
                get_param: DnsServers
              members:
                - type: interface
                  name: nic3
                  primary: true
                - type: interface
                  name: nic4

            - type: vlan
              vlan_id:
                get_param: InternalApiNetworkVlanID
              device: bond_api
              addresses:
                - ip_netmask:
                    get_param: InternalApiIpSubnet

            - type: vlan
              vlan_id:
                get_param: TenantNetworkVlanID

```

```

    device: bond_api
    addresses:
      - ip_netmask:
          get_param: TenantIpSubnet

  - type: vlan
    vlan_id:
      get_param: StorageNetworkVlanID
    device: bond_api
    addresses:
      - ip_netmask:
          get_param: StorageIpSubnet

  - type: interface
    name: ens2f0
    mtu: 9000
    use_dhcp: false
    defroute: false
    nm_controlled: true
    hotplug: true

  - type: interface
    name: ens2f1
    mtu: 9000
    use_dhcp: false
    defroute: false
    nm_controlled: true
    hotplug: true

```

```

outputs:
  OS::stack_id:
    description: The OsNetConfigImpl resource.
    value:
      get_resource: OsNetConfigImpl

```

C.1.7. overcloud_deploy.sh

```

#!/bin/bash

openstack overcloud deploy \
--templates \
-r /home/stack/ospd-12-sriov-dpdk-heterogeneous-cluster/roles_data.yaml \
-e /usr/share/openstack-tripleo-heat-templates/environments/network-
isolation.yaml \
-e /usr/share/openstack-tripleo-heat-templates/environments/host-config-
and-reboot.yaml \
-e /usr/share/openstack-tripleo-heat-templates/environments/neutron-
sriov.yaml \
-e /usr/share/openstack-tripleo-heat-templates/environments/neutron-ovs-
dpdk.yaml \
-e /home/stack/ospd-12-vlan-sriov-two-ports-ctlplane-bonding/docker-
images.yaml \
-e /home/stack/ospd-12-sriov-dpdk-heterogeneous-cluster/network-
environment.yaml \
--log-file overcloud_install.log &> overcloud_install.log

```

APPENDIX D. DIFFERENT INTERFACES ON SAME COMPUTE NODE YAML FILES

This section provides sample YAML files as a reference for adding SR-IOV and DPDK interfaces on the same compute node.

D.1. SAMPLE SR-IOV AND DPDK ON THE SAME COMPUTE NODE YAML FILES

D.1.1. roles_data.yaml

```
#####
#####
# File generated by TripleO
#####
#####
#####
#####
# Role: Controller
#
#####
#####
- name: Controller
  description: |
    Controller role that has all the controller services loaded and handles
    Database, Messaging and Network functions.
  CountDefault: 1
  tags:
    - primary
    - controller
  networks:
    - External
    - InternalApi
    - Storage
    - StorageMgmt
    - Tenant
  HostnameFormatDefault: 'controller-%index%'
  # Deprecated & backward-compatible values (FIXME: Make parameters
consistent)
  # Set uses_deprecated_params to True if any deprecated params are used.
  uses_deprecated_params: True
  deprecated_param_extraconfig: 'controllerExtraConfig'
  deprecated_param_flavor: 'OvercloudControlFlavor'
  deprecated_param_image: 'controllerImage'
  ServicesDefault:
    - OS::TripleO::Services::AodhApi
    - OS::TripleO::Services::AodhEvaluator
    - OS::TripleO::Services::AodhListener
    - OS::TripleO::Services::AodhNotifier
    - OS::TripleO::Services::AuditD
    - OS::TripleO::Services::BarbicanApi
    - OS::TripleO::Services::CACerts
    - OS::TripleO::Services::CeilometerAgentCentral
```

```

- OS::Triple0::Services::CeilometerAgentNotification
# FIXME: This service was disabled in Pike and this entry should be
removed
# in Queens.
- OS::Triple0::Services::CeilometerApi
- OS::Triple0::Services::CeilometerCollector
- OS::Triple0::Services::CeilometerExpirer
- OS::Triple0::Services::CephExternal
- OS::Triple0::Services::CephMds
- OS::Triple0::Services::CephMon
- OS::Triple0::Services::CephRbdMirror
- OS::Triple0::Services::CephRgw
- OS::Triple0::Services::CertmongerUser
- OS::Triple0::Services::CinderApi
- OS::Triple0::Services::CinderBackendDellPs
- OS::Triple0::Services::CinderBackendDellSc
- OS::Triple0::Services::CinderBackendDellEMCUnity
- OS::Triple0::Services::CinderBackendDellEMCVMAXISCSI
- OS::Triple0::Services::CinderBackendNetApp
- OS::Triple0::Services::CinderBackendScaleIO
- OS::Triple0::Services::CinderBackendVRTSHyperScale
- OS::Triple0::Services::CinderBackup
- OS::Triple0::Services::CinderHPELeftHandISCSI
- OS::Triple0::Services::CinderScheduler
- OS::Triple0::Services::CinderVolume
- OS::Triple0::Services::Clustercheck
- OS::Triple0::Services::Collectd
- OS::Triple0::Services::Congress
- OS::Triple0::Services::Docker
- OS::Triple0::Services::Ec2Api
- OS::Triple0::Services::Etcd
- OS::Triple0::Services::ExternalSwiftProxy
- OS::Triple0::Services::FluentdClient
- OS::Triple0::Services::GlanceApi
- OS::Triple0::Services::GnocchiApi
- OS::Triple0::Services::GnocchiMetricd
- OS::Triple0::Services::GnocchiStatsd
- OS::Triple0::Services::HAproxy
- OS::Triple0::Services::HeatApi
- OS::Triple0::Services::HeatApiCfn
- OS::Triple0::Services::HeatApiCloudwatch
- OS::Triple0::Services::HeatEngine
- OS::Triple0::Services::Horizon
- OS::Triple0::Services::IronicApi
- OS::Triple0::Services::IronicConductor
- OS::Triple0::Services::Isctsid
- OS::Triple0::Services::Keepalived
- OS::Triple0::Services::Kernel
- OS::Triple0::Services::Keystone
- OS::Triple0::Services::ManilaApi
- OS::Triple0::Services::ManilaBackendCephFs
- OS::Triple0::Services::ManilaBackendGeneric
- OS::Triple0::Services::ManilaBackendIsilon
- OS::Triple0::Services::ManilaBackendNetapp
- OS::Triple0::Services::ManilaBackendUnity
- OS::Triple0::Services::ManilaBackendVNX

```

- OS::Triple0::Services::ManilaBackendVMAX
- OS::Triple0::Services::ManilaScheduler
- OS::Triple0::Services::ManilaShare
- OS::Triple0::Services::Memcached
- OS::Triple0::Services::MongoDb
- OS::Triple0::Services::MySQL
- OS::Triple0::Services::MySQLClient
- OS::Triple0::Services::NeutronApi
- OS::Triple0::Services::NeutronBgpVpnApi
- OS::Triple0::Services::NeutronCorePlugin
- OS::Triple0::Services::NeutronDhcpAgent
- OS::Triple0::Services::NeutronL2gwAgent
- OS::Triple0::Services::NeutronL2gwApi
- OS::Triple0::Services::NeutronL3Agent
- OS::Triple0::Services::NeutronLbaasv2Agent
- OS::Triple0::Services::NeutronLinuxbridgeAgent
- OS::Triple0::Services::NeutronMetadataAgent
- OS::Triple0::Services::NeutronML2FujitsuCfab
- OS::Triple0::Services::NeutronML2FujitsuFossw
- OS::Triple0::Services::NeutronOvsAgent
- OS::Triple0::Services::NeutronVppAgent
- OS::Triple0::Services::NovaApi
- OS::Triple0::Services::NovaConductor
- OS::Triple0::Services::NovaConsoleauth
- OS::Triple0::Services::NovaIronic
- OS::Triple0::Services::NovaMetadata
- OS::Triple0::Services::NovaPlacement
- OS::Triple0::Services::NovaScheduler
- OS::Triple0::Services::NovaVncProxy
- OS::Triple0::Services::Ntp
- OS::Triple0::Services::ContainersLogrotateCronD
- OS::Triple0::Services::OctaviaApi
- OS::Triple0::Services::OctaviaHealthManager
- OS::Triple0::Services::OctaviaHousekeeping
- OS::Triple0::Services::OctaviaWorker
- OS::Triple0::Services::OpenDaylightApi
- OS::Triple0::Services::OpenDaylightOvs
- OS::Triple0::Services::OVNDBs
- OS::Triple0::Services::OVNController
- OS::Triple0::Services::Pacemaker
- OS::Triple0::Services::PankoApi
- OS::Triple0::Services::RabbitMQ
- OS::Triple0::Services::Redis
- OS::Triple0::Services::SaharaApi
- OS::Triple0::Services::SaharaEngine
- OS::Triple0::Services::Securetty
- OS::Triple0::Services::SensuClient
- OS::Triple0::Services::Snmp
- OS::Triple0::Services::Sshd
- OS::Triple0::Services::SwiftProxy
- OS::Triple0::Services::SwiftRingBuilder
- OS::Triple0::Services::SwiftStorage
- OS::Triple0::Services::Tacker
- OS::Triple0::Services::Timezone
- OS::Triple0::Services::TripleoFirewall
- OS::Triple0::Services::TripleoPackages

```

- OS::Triple0::Services::Tuned
- OS::Triple0::Services::Vpp
- OS::Triple0::Services::Zaqar
#####
#####
# Role: ComputeOvsDpdkSriov
#
#####
#####
- name: ComputeOvsDpdkSriov
  description: |
    Compute OvS DPDK Sriov Role
  CountDefault: 1
  networks:
    - InternalApi
    - Tenant
    - Storage
  HostnameFormatDefault: 'computeovsdpdkSriov-%index%'
  disable_upgrade_deployment: True
  ServicesDefault:
    - OS::Triple0::Services::AuditD
    - OS::Triple0::Services::CACerts
    - OS::Triple0::Services::CephClient
    - OS::Triple0::Services::CephExternal
    - OS::Triple0::Services::CertmongerUser
    - OS::Triple0::Services::Collectd
    - OS::Triple0::Services::ComputeCeilometerAgent
    - OS::Triple0::Services::ComputeNeutronCorePlugin
    - OS::Triple0::Services::ComputeNeutronL3Agent
    - OS::Triple0::Services::ComputeNeutronMetadataAgent
    - OS::Triple0::Services::ComputeNeutronOvsDpdk
    - OS::Triple0::Services::Docker
    - OS::Triple0::Services::FluentdClient
    - OS::Triple0::Services::Iscsid
    - OS::Triple0::Services::Kernel
    - OS::Triple0::Services::MySQLClient
    - OS::Triple0::Services::NovaCompute
    - OS::Triple0::Services::NovaLibvirt
    - OS::Triple0::Services::NovaMigrationTarget
    - OS::Triple0::Services::NeutronLinuxbridgeAgent
    - OS::Triple0::Services::NeutronSriovAgent
    - OS::Triple0::Services::NeutronVppAgent
    - OS::Triple0::Services::Ntp
    - OS::Triple0::Services::ContainersLogrotateCronD
    - OS::Triple0::Services::OpenDaylightOvs
    - OS::Triple0::Services::Securetty
    - OS::Triple0::Services::SensuClient
    - OS::Triple0::Services::Snmp
    - OS::Triple0::Services::Sshd
    - OS::Triple0::Services::Timezone
    - OS::Triple0::Services::TripleoFirewall
    - OS::Triple0::Services::TripleoPackages
    - OS::Triple0::Services::Tuned
    - OS::Triple0::Services::Vpp
    - OS::Triple0::Services::OVNController

```


D.1.2. post-install.yaml

```

heat_template_version: 2014-10-16

description: >
  Example extra config for post-deployment

parameters:
  servers:
    type: json

resources:

  ExtraDeployments:
    type: OS::Heat::StructuredDeployments
    properties:
      servers: {get_param: servers}
      config: {get_resource: ExtraConfig}
      actions: ['CREATE', 'UPDATE']

  ExtraConfig:
    type: OS::Heat::SoftwareConfig
    properties:
      group: script
      config: |
        #!/bin/bash
        set -x
        function tuned_service_dependency() {
          tuned_service=/usr/lib/systemd/system/tuned.service
          grep -q "network.target" $tuned_service
          if [ "$?" -eq 0 ]; then
            sed -i '/After=.*s/network.target//g' $tuned_service
          fi
          grep -q "Before=.*network.target" $tuned_service
          if [ ! "$?" -eq 0 ]; then
            grep -q "Before=.*" $tuned_service
            if [ "$?" -eq 0 ]; then
              sed -i 's/^\(Before=.*\)\/\1 network.target
openvswitch.service/g' $tuned_service
            else
              sed -i '/After/i Before=network.target
openvswitch.service' $tuned_service
            fi
          fi
        }

        if hiera -c /etc/puppet/hiera.yaml service_names | grep -q
neutron_ovs_dpdk_agent; then
          tuned_service_dependency
        fi

```

D.1.3. network.environment.yaml

```

resource_registry:
  # Specify the relative/absolute path to the config files you want to use

```

for override the default.

```
OS::TripleO::ComputeOvsDpdkSriov::Net::SoftwareConfig: nic-
configs/computeovsdpdkSriov.yaml
```

```
OS::TripleO::Controller::Net::SoftwareConfig: nic-
configs/controller.yaml
```

```
OS::TripleO::NodeExtraConfigPost: post-install.yaml
```

parameter_defaults:

Customize all these values to match the local environment

```
InternalApiNetCidr: 10.10.10.0/24
```

```
TenantNetCidr: 10.10.2.0/24
```

```
StorageNetCidr: 10.10.3.0/24
```

```
StorageMgmtNetCidr: 10.10.4.0/24
```

```
ExternalNetCidr: 172.20.12.112/28
```

CIDR subnet mask length for provisioning network

```
ControlPlaneSubnetCidr: '24'
```

```
InternalApiAllocationPools: [{'start': '10.10.10.10', 'end':
'10.10.10.200'}]
```

```
TenantAllocationPools: [{'start': '10.10.2.100', 'end': '10.10.2.200'}]
```

```
StorageAllocationPools: [{'start': '10.10.3.100', 'end': '10.10.3.200'}]
```

```
StorageMgmtAllocationPools: [{'start': '10.10.4.100', 'end':
'10.10.4.200'}]
```

Use an External allocation pool which will leave room for floating IPs

```
ExternalAllocationPools: [{'start': '172.20.12.114', 'end':
'172.20.12.125'}]
```

Set to the router gateway on the external network

```
ExternalInterfaceDefaultRoute: 172.20.12.126
```

Gateway router for the provisioning network (or Undercloud IP)

```
ControlPlaneDefaultRoute: 192.168.24.1
```

Generally the IP of the Undercloud

```
EC2MetadataIp: 192.168.24.1
```

```
InternalApiNetworkVlanID: 10
```

```
TenantNetworkVlanID: 11
```

```
StorageNetworkVlanID: 12
```

```
StorageMgmtNetworkVlanID: 13
```

```
ExternalNetworkVlanID: 14
```

Define the DNS servers (maximum 2) for the overcloud nodes

```
DnsServers: ["8.8.8.8", "8.8.4.4"]
```

*# May set to br-ex if using floating IPs only on native VLAN on bridge
br-ex*

```
NeutronExternalNetworkBridge: ""
```

*# The tunnel type for the tenant network (vxlan or gre). Set to '' to
disable tunneling.*

```
NeutronTunnelTypes: ''
```

The tenant network type for Neutron (vlan or vxlan).

```
NeutronNetworkType: 'vlan'
```

The OVS logical->physical bridge mappings to use.

```
NeutronBridgeMappings: 'tenant_dpdk:br-link0,tenant:br-link1'
```

The Neutron ML2 and OpenVSwitch vlan mapping range to support.

```
NeutronNetworkVLANRanges: 'tenant_dpdk:22:22,tenant:25:25'
```

Nova flavor to use.

```
OvercloudControllerFlavor: controller
```

```
OvercloudComputeOvsDpdkSriovFlavor: compute
```

#Number of nodes to deploy.

```
ControllerCount: 1
```

```
ComputeOvsDpdkSriovCount: 1
```

```

# NTP server configuration.
NtpServer: clock.redhat.com

#####
# OVS DPDK configuration #
# #####
ComputeOvsDpdkSriovParameters:
  KernelArgs: default_hugepagesz=1GB hugepagesz=1G hugepages=32 iommu=pt
intel_iommu=on
  TunedProfileName: "cpu-partitioning"
  IsolCpusList:
"1,2,3,4,5,6,7,9,10,17,18,19,20,21,22,23,11,12,13,14,15,25,26,27,28,29,30,
31"
  NovaVcpuPinSet:
['2,3,4,5,6,7,18,19,20,21,22,23,10,11,12,13,14,15,26,27,28,29,30,31']
  NovaReservedHostMemory: 4096
  OvsDpdkSocketMemory: "1024,1024"
  OvsDpdkMemoryChannels: "4"
  OvsDpdkCoreList: "0,16,8,24"
  OvsPmdCoreList: "1,17,9,25"

# MTU global configuration
NeutronGlobalPhysnetMtu: 9000
# DHCP provide metadata route to VM.
NeutronEnableIsolatedMetadata: true
# DHCP always provides metadata route to VM.
NeutronEnableForceMetadata: true
# Configure the classname of the firewall driver to use for implementing
security groups.
NeutronOVSFirewallDriver: openvswitch

#SR-IOV params
NeutronMechanismDrivers: ['openvswitch','sriovnicswitch']
NovaSchedulerDefaultFilters:
['RetryFilter','AvailabilityZoneFilter','RamFilter','ComputeFilter','Compu
teCapabilitiesFilter','ImagePropertiesFilter','ServerGroupAntiAffinityFilt
er','ServerGroupAffinityFilter','PciPassthroughFilter','NUMATopologyFilter
']
NovaSchedulerAvailableFilters:
["nova.scheduler.filters.all_filters","nova.scheduler.filters.pci_passtro
ugh_filter.PciPassthroughFilter"]
NeutronSupportedPCIVendorDevs: ['8086:154d', '8086:10ed']
NovaPCIPassthrough:
- devname: "ens2f1"
  physical_network: "tenant"

NeutronPhysicalDevMappings: "tenant:ens2f1"
NeutronSriovNumVFs: "ens2f1:5"

```

D.1.4. controller.yaml

```

heat_template_version: pike

description: >
  Software Config to drive os-net-config to configure VLANs for the

```

controller role.

parameters:

```
ControlPlaneIp:
  default: ''
  description: IP address/subnet on the ctlplane network
  type: string
ExternalIpSubnet:
  default: ''
  description: IP address/subnet on the external network
  type: string
InternalApiIpSubnet:
  default: ''
  description: IP address/subnet on the internal API network
  type: string
StorageNetworkVlanID:
  default: 30
  description: Vlan ID for the storage network traffic.
  type: number
StorageMgmtNetworkVlanID:
  default: 40
  description: Vlan ID for the storage mgmt network traffic.
  type: number
StorageIpSubnet:
  default: ''
  description: IP address/subnet on the storage network
  type: string
StorageMgmtIpSubnet:
  default: ''
  description: IP address/subnet on the storage mgmt network
  type: string
TenantIpSubnet:
  default: ''
  description: IP address/subnet on the tenant network
  type: string
ManagementIpSubnet: # Only populated when including
environments/network-management.yaml
  default: ''
  description: IP address/subnet on the management network
  type: string
ExternalNetworkVlanID:
  default: ''
  description: Vlan ID for the external network traffic.
  type: number
InternalApiNetworkVlanID:
  default: ''
  description: Vlan ID for the internal_api network traffic.
  type: number
TenantNetworkVlanID:
  default: ''
  description: Vlan ID for the tenant network traffic.
  type: number
ManagementNetworkVlanID:
  default: 23
  description: Vlan ID for the management network traffic.
  type: number
```

```

ExternalInterfaceDefaultRoute:
  default: ''
  description: default route for the external network
  type: string
ControlPlaneSubnetCidr: # Override this via parameter_defaults
  default: '24'
  description: The subnet CIDR of the control plane network.
  type: string
ControlPlaneDefaultRoute: # Override this via parameter_defaults
  description: The default route of the control plane network.
  type: string
DnsServers: # Override this via parameter_defaults
  default: []
  description: A list of DNS servers (2 max for some implementations)
that will be added to resolv.conf.
  type: comma_delimited_list
EC2MetadataIp: # Override this via parameter_defaults
  description: The IP address of the EC2 metadata server.
  type: string

resources:
  OsNetConfigImpl:
    type: OS::Heat::SoftwareConfig
    properties:
      group: script
      config:
        str_replace:
          template:
            get_file: /usr/share/openstack-tripleo-heat-
templates/network/scripts/run-os-net-config.sh
        params:
          $network_config:
            network_config:
              - type: interface
                name: nic1
                use_dhcp: false
                defroute: false

              - type: linux_bond
                name: bond_api
                bonding_options: "mode=active-backup"
                use_dhcp: false
                dns_servers:
                  get_param: DnsServers
                addresses:
                  - ip_netmask:
                      list_join:
                        - /
                        - - get_param: ControlPlaneIp
                          - get_param: ControlPlaneSubnetCidr
                routes:
                  - ip_netmask: 169.254.169.254/32
                    next_hop:
                      get_param: EC2MetadataIp
                members:
                  - type: interface

```

```

        name: nic2
        primary: true
    - type: interface
      name: nic3

    - type: vlan
      vlan_id:
        get_param: InternalApiNetworkVlanID
      device: bond_api
      addresses:
        - ip_netmask:
            get_param: InternalApiIpSubnet

    - type: vlan
      vlan_id:
        get_param: TenantNetworkVlanID
      device: bond_api
      addresses:
        - ip_netmask:
            get_param: TenantIpSubnet

    - type: vlan
      vlan_id:
        get_param: StorageNetworkVlanID
      device: bond_api
      addresses:
        - ip_netmask:
            get_param: StorageIpSubnet

    - type: vlan
      vlan_id:
        get_param: StorageMgmtNetworkVlanID
      device: bond_api
      addresses:
        - ip_netmask:
            get_param: StorageMgmtIpSubnet

    - type: vlan
      vlan_id:
        get_param: ExternalNetworkVlanID
      device: bond_api
      addresses:
        - ip_netmask:
            get_param: ExternalIpSubnet
      routes:
        - default: true
          next_hop:
            get_param: ExternalInterfaceDefaultRoute

    - type: ovs_bridge
      name: br-link0
      use_dhcp: false
      members:
        - type: interface
          name: nic4
          mtu: 9000

```

```

        - type: ovs_bridge
          name: br-link1
          use_dhcp: false
          members:
            - type: interface
              name: nic5
              mtu: 9000

outputs:
  OS::stack_id:
    description: The OsNetConfigImpl resource.
    value:
      get_resource: OsNetConfigImpl

```

D.1.5. compute-ovsdpdksriov.yaml

```

heat_template_version: pike

description: >
  Software Config to drive os-net-config to configure VLANs for the
  compute role.

parameters:
  ControlPlaneIp:
    default: ''
    description: IP address/subnet on the ctlplane network
    type: string
  ExternalIpSubnet:
    default: ''
    description: IP address/subnet on the external network
    type: string
  InternalApiIpSubnet:
    default: ''
    description: IP address/subnet on the internal API network
    type: string
  TenantIpSubnet:
    default: ''
    description: IP address/subnet on the tenant network
    type: string
  ManagementIpSubnet: # Only populated when including
environments/network-management.yaml
    default: ''
    description: IP address/subnet on the management network
    type: string
  InternalApiNetworkVlanID:
    default: ''
    description: Vlan ID for the internal_api network traffic.
    type: number
  TenantNetworkVlanID:
    default: ''
    description: Vlan ID for the tenant network traffic.
    type: number
  ManagementNetworkVlanID:
    default: 23

```

```

    description: Vlan ID for the management network traffic.
    type: number
StorageNetworkVlanID:
    default: 30
    description: Vlan ID for the storage network traffic.
    type: number
StorageIpSubnet:
    default: ''
    description: IP address/subnet on the storage network
    type: string
StorageMgmtIpSubnet:
    default: ''
    description: IP address/subnet on the storage mgmt network
    type: string
ControlPlaneSubnetCidr: # Override this via parameter_defaults
    default: '24'
    description: The subnet CIDR of the control plane network.
    type: string
ControlPlaneDefaultRoute: # Override this via parameter_defaults
    description: The default route of the control plane network.
    type: string
DnsServers: # Override this via parameter_defaults
    default: []
    description: A list of DNS servers (2 max for some implementations)
that will be added to resolv.conf.
    type: comma_delimited_list
EC2MetadataIp: # Override this via parameter_defaults
    description: The IP address of the EC2 metadata server.
    type: string
ExternalInterfaceDefaultRoute:
    default: ''
    description: default route for the external network
    type: string

resources:
  OsNetConfigImpl:
    type: OS::Heat::SoftwareConfig
    properties:
      group: script
      config:
        str_replace:
          template:
            get_file: /usr/share/openstack-tripleo-heat-
templates/network/scripts/run-os-net-config.sh
          params:
            $network_config:
              network_config:
                - type: interface
                  name: nic1
                  use_dhcp: false
                  defroute: false

                - type: interface
                  name: nic2
                  use_dhcp: false
                  addresses:

```



```

- ip_netmask:
  list_join:
    - /
    - - get_param: ControlPlaneIp
      - get_param: ControlPlaneSubnetCidr
routes:
- ip_netmask: 169.254.169.254/32
  next_hop:
    get_param: EC2MetadataIp
- default: true
  next_hop:
    get_param: ControlPlaneDefaultRoute

- type: linux_bond
  name: bond_api
  bonding_options: "mode=active-backup"
  use_dhcp: false
  dns_servers:
    get_param: DnsServers
  members:
    - type: interface
      name: nic3
      primary: true
    - type: interface
      name: nic4

- type: vlan
  vlan_id:
    get_param: InternalApiNetworkVlanID
  device: bond_api
  addresses:
    - ip_netmask:
        get_param: InternalApiIpSubnet

- type: vlan
  vlan_id:
    get_param: TenantNetworkVlanID
  device: bond_api
  addresses:
    - ip_netmask:
        get_param: TenantIpSubnet

- type: vlan
  vlan_id:
    get_param: StorageNetworkVlanID
  device: bond_api
  addresses:
    - ip_netmask:
        get_param: StorageIpSubnet

- type: ovs_user_bridge
  name: br-link0
  use_dhcp: false
  members:
    - type: ovs_dpdk_port
      name: dpdk0

```

```

        mtu: 9000
        rx_queue: 2
        members:
          - type: interface
            name: nic5

      - type: interface
        name: ens2f1
        mtu: 9000
        use_dhcp: false
        defroute: false
        nm_controlled: true
        hotplug: true

outputs:
  OS::stack_id:
    description: The OsNetConfigImpl resource.
    value:
      get_resource: OsNetConfigImpl

```

D.1.6. overcloud_deploy.sh

```

#!/bin/bash

openstack overcloud deploy \
--templates \
-r /home/stack/ospd-12-vlan-dpdk-sriov-two-ports-ctlplane-
bonding/roles_data.yaml \
-e /usr/share/openstack-tripleo-heat-templates/environments/network-
isolation.yaml \
-e /usr/share/openstack-tripleo-heat-templates/environments/host-config-
and-reboot.yaml \
-e /usr/share/openstack-tripleo-heat-templates/environments/neutron-ovs-
dpdk.yaml \
-e /usr/share/openstack-tripleo-heat-templates/environments/neutron-
sriov.yaml \
-e /home/stack/ospd-12-vlan-dpdk-sriov-two-ports-ctlplane-bonding/docker-
images.yaml \
-e /home/stack/ospd-12-vlan-dpdk-sriov-two-ports-ctlplane-bonding/network-
environment.yaml \
--log-file overcloud_install.log &> overcloud_install.log

```

APPENDIX E. REVISION HISTORY

Revision 1.2-0

August 17 2018

Marked step as optional: 'map the physical network to the logical bridge'.

Revision 1.1-0

July 31 2018

Updated network creation steps to use OSC parameters. Added description of BIOS settings.