NETRONOME

Netronome Agilio® SmartNIC

Agilio Open vSwitch

User Guide for

Red Hat OpenStack Platform 13

3 July 2018

Revision: 1.1
3 July 2018

1. Preface

1.1. Introduction

This guide details the configuration of Red Hat OpenStack Platform 13 to use the Open vSwitch TC offload features of the Agilio CX SmartNIC series.

1.2. List of Acronyms

Acronym	
CLI	Command-line Interface
COTS	Commercial Off-The-Shelf
IOMMU	Input/Output Memory Management Unit
NFP	Network Flow Processor
OCP	Open Compute Project
OS	Operating System
OVS	Open Virtual Switch
PCI	Peripheral Component Interconnect
RHEL	Red Hat Enterprise Linux
RHOSP	Red Hat OpenStack Platform
SR-IOV	Single-Root Input/Output Virtualization
TC	The Linux Traffic Control Subsystem
VFIO	Virtual Function Input/Output

2. The Agilio® SmartNIC Architecture

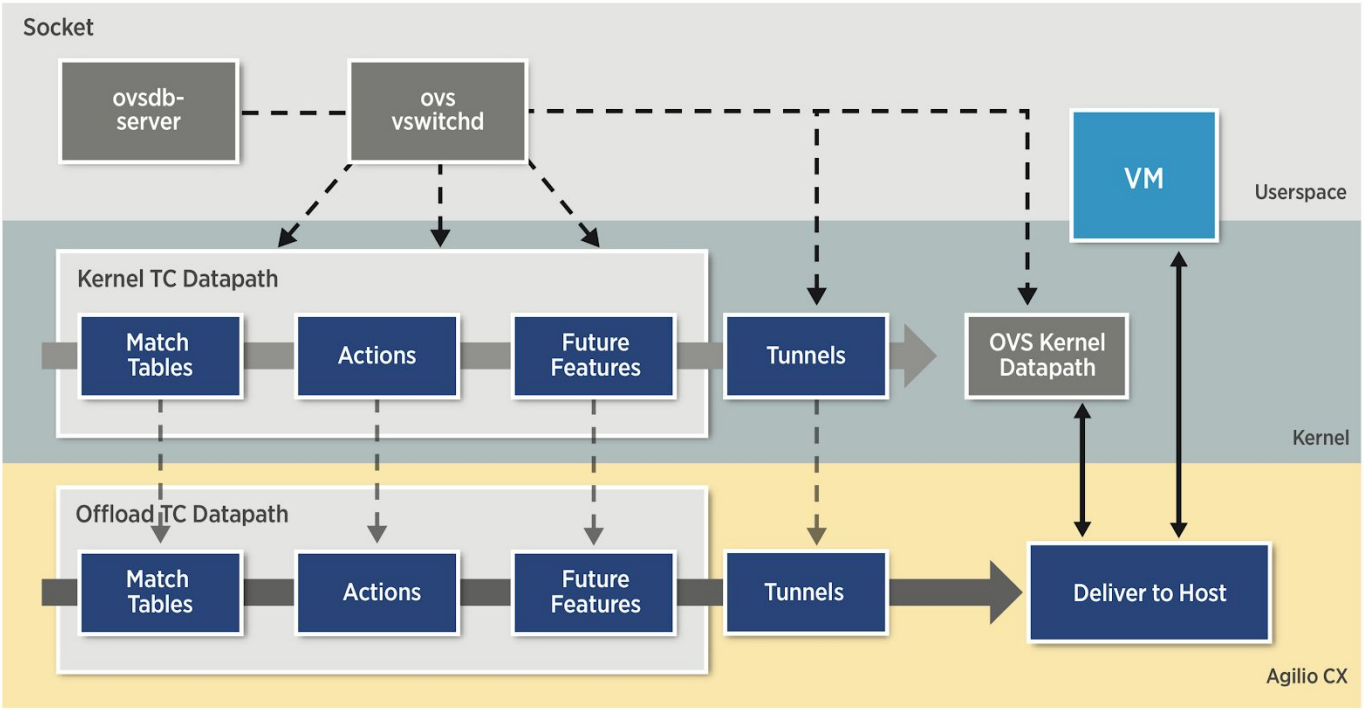

Figure 1. The conceptual architecture of the Agilio SmartNIC

The Agilio® CX SmartNICs are based on the NFP-4000 and are available in low profile PCIe and OCP v2 NIC form factors suitable for COTS servers. They feature a 60 core processor with eight cooperatively multithreaded threads per core. The flow processing cores have an instruction set that is optimized for networking. The OVS datapath firmware can be enabled without a server reboot.

3. Required Versions

3.1. Hardware

The minimum SmartNIC NVRAM version is: `020025.020025.02006e`. Refer to the Agilio Open vSwitch User Guide on how to determine which SmartNIC NVRAM version has been installed on the card.

3.2. Software

This guide covers Red Hat Enterprise Linux Server release 7.5 and Red Hat OpenStack Platform 13 (based on OpenStack Queens).

4. Initial Hardware Installation

4.1. Validation

It is recommended that the SmartNIC be installed and validated on a representative compute node, with a copy of RHEL 7.5. Follow the instructions in the Agilio Open vSwitch User Guide to select the TC offload firmware, and determine the relevant netdev interfaces presented. As an example, this guide will use the following example throughout:

Role	**Netdev**
Physical Function	`ens3`
Physical Port 0 Representor	`ens3np0`
Physical Port 1 Representor	`ens3np1`

Note: Some platforms may renumber the PCI slot when enabling and disabling features like IOMMU passthrough. Take care to ensure the BIOS settings and kernel command line corresponds to the intended production configuration of the node, in particular, passing `intel_iommu=on` as a kernel boot parameter.

4.2. Identification

Consult the user guide for information on how to identify the serial number and assembly ID on the physical card. In a running system, the assembly ID and serial number of a device may be read via the ethtool debug interface. The following shell snippet illustrates this method for the netdev `ens3np0`:

```
#!/bin/sh
DEVICE=ens3np0
ethtool -W ${DEVICE} 0
DEBUG=$(ethtool -w ${DEVICE} data /dev/stdout | strings)
SERIAL=$(echo "${DEBUG}" | grep "^SN:")
ASSY=$(echo ${SERIAL} | grep -oE AMDA[0-9]{4})
echo Serial: ${SERIAL}
echo Assembly: ${ASSY}
```

5. Director Configuration

Note: In this section, it is assumed that that the initial setup of the undercloud on the Director has been completed. All commands are assumed to be run as the `stack` user with `~/stackrc` sourced:

```
[stack@director ~]$ source ~/stackrc
(undercloud) [stack@director ~]$
```

5.1. Planning: VXLAN Overlay Topology

The following example illustrates the recommended topology to employ VXLAN overlays on a tenant network of a Compute SR-IOV Node:

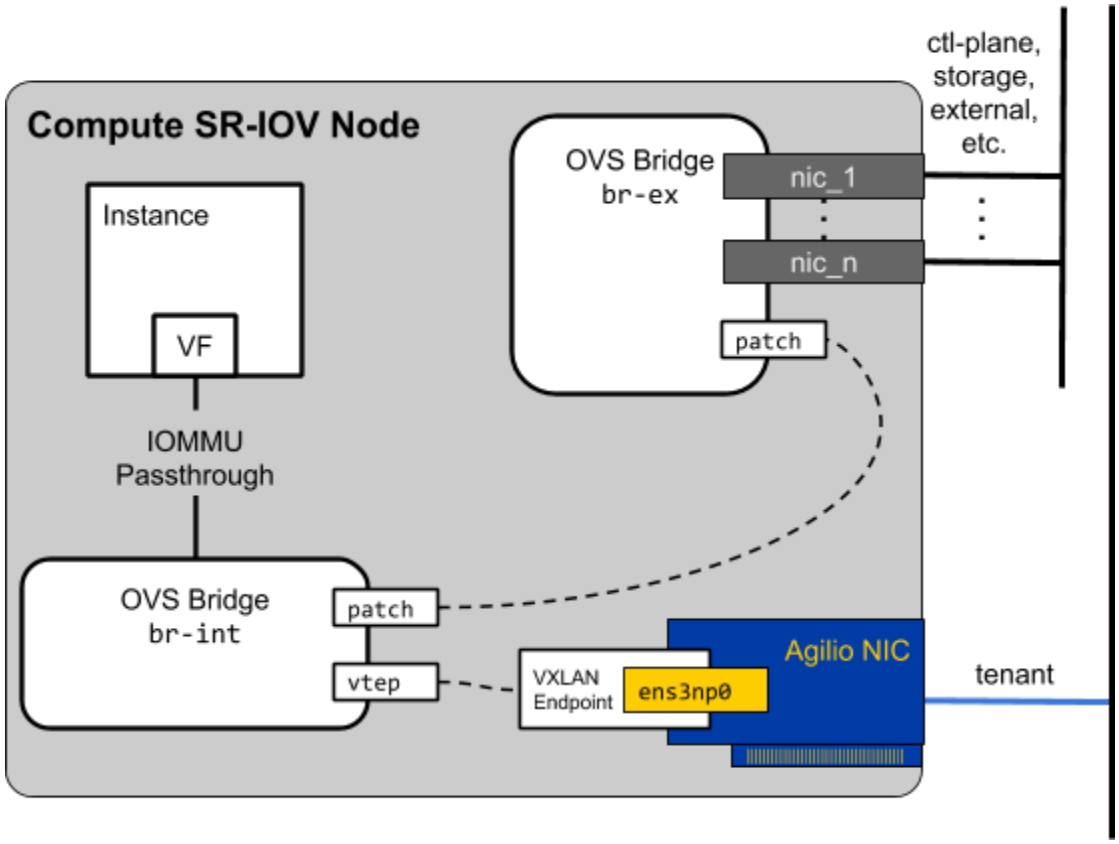

Figure 2. An example network configuration for a Compute SR-IOV node.

Note: To ensure offloadable flows are generated, assign the IP address of the VXLAN endpoint of the underlay network directly to the physical port representor.

5.2. Create a Compute SR-IOV Role

Consult the section on "Composable Services and Custom Roles" in the Red Hat OpenStack Advanced Overcloud Customization guide on creating a custom `ComputeSriov` role. For example, the following command will create a `roles_data.yaml` file with a `Controller`, `ComputeSriov` and `Compute` role:

```
openstack overcloud roles generate \
    --roles-path /usr/share/openstack-tripleo-heat-templates/roles \
    -o roles_data.yaml \
    Controller ComputeSriov Compute
```

This file can then be further customized. At this point, note the template for the hostname in the `HostnameFormatDefault` field. If this is customized, also take care to customize it in the firmware selection configuration described in the next section. This is an example of a `ComputeSriov` role:

```
###############################################################################
# Role: ComputeSriov                                                          #
###############################################################################
- name: ComputeSriov
  description: |
    Compute SR-IOV Role
  CountDefault: 1
  networks:
    - InternalApi
    - Tenant
    - Storage
  HostnameFormatDefault: '%stackname%-computesriov-%index%'
  disable_upgrade_deployment: True
  ServicesDefault:
    - OS::TripleO::Services::Aide
    - 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::Fluentd
    - OS::TripleO::Services::Ipsec
```

```
    - OS::TripleO::Services::Iscsid
    - OS::TripleO::Services::Kernel
    - OS::TripleO::Services::LoginDefs
    - OS::TripleO::Services::MySQLClient
    - OS::TripleO::Services::NeutronBgpVpnBagpipe
    - 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::Rhsm
    - OS::TripleO::Services::RsyslogSidecar
    - OS::TripleO::Services::Securetty
    - OS::TripleO::Services::SensuClient
    - OS::TripleO::Services::SkydiveAgent
    - OS::TripleO::Services::Snmp
    - OS::TripleO::Services::Sshd
    - OS::TripleO::Services::Timezone
    - OS::TripleO::Services::TripleoFirewall
    - OS::TripleO::Services::TripleoPackages
    - OS::TripleO::Services::Vpp
    - OS::TripleO::Services::OVNController
    - OS::TripleO::Services::OVNMetadataAgent
    - OS::TripleO::Services::Ptp
```

5.3. Create or modify `firstboot.yaml`

In order to select the TC Offload firmware, a userdata software config resource needs to be applied on node commissioning. The recommended method is to create a `firstboot.yaml` file with the following contents:

```
heat_template_version: queens

description: >
  Firstboot actions (set firmware on compute nodes)

resources:
  userdata:
    type: OS::Heat::MultipartMime
    properties:
      parts:
      - config: {get_resource: select_firmware}

  select_firmware:
    type: OS::Heat::SoftwareConfig
    properties:
      config:
        str_replace:
          template: |
            #!/bin/bash
            DEVICE=$netdev
            APP=flower
            FWDIR=/lib/firmware/netronome
            DRACUT_CONF=/etc/dracut.conf.d/98-nfp-firmware.conf
            DEBUG_FILE=/var/run/cloud-init/ethtool.dbg.${DEVICE}
            if [[ `hostname` = *"$rolename"* ]]; then
                if [ -e /sys/class/net/${DEVICE} ]; then
                    ethtool -W ${DEVICE} 0
                    ethtool -w ${DEVICE} data ${DEBUG_FILE}
                    SERIAL=`strings ${DEBUG_FILE} | grep "^SN:"`
                    ASSY=`echo ${SERIAL} | grep -oE AMDA[0-9]{4}`
                    SYSPATH=`readlink -e /sys/class/net/${DEVICE}/device`
                    PCIADDR=`basename ${SYSPATH}`
                    FW="${FWDIR}/pci-${PCIADDR}.nffw"
                    ln -sf "${APP}/nic_${ASSY}.nffw" "${FW}"
                    rmmod nfp
                    modprobe nfp
                    echo "install_items+=\" ${FW} \"" >"${DRACUT_CONF}"
                    dracut -f
                fi
            fi
```

```
          params:
              $netdev: ens3np0
              $rolename: computesriov

outputs:
  OS::stack_id:
    value: {get_resource: userdata}
```

Important Notes:

- Ensure that the netdev name (here, `ens3np0`) is replaced with the correct value for your system.
- Ensure that the role name (here, `computesriov`) matches the template generated in the previous section.
- If other software configurations need to be applied at the same config hook (`OS::TripleO::NodeUserData`), they will need to be manually merged.

5.4. Set up `nic-configs/compute-sriov.yaml`

The network configuration configuration description of the `ComputeSriov` node should be updated according to the network isolation configuration. The following snippet illustrates one possible configuration:

```
heat_template_version: queens

parameters:
  ...snip...

resources:
  OsNetConfigImpl:
    type: OS::Heat::SoftwareConfig
    properties:
      group: script
      config:
        str_replace:
          template:
            get_file: /path/to/run-os-net-config.sh
          params:
            $network_config:
              network_config:
              ...snip...other...interface...config...
              - type: interface
                # Enable the PF interface but don't DHCP on it
                name: ens3
                use_dhcp: false
              - type: interface
```

```
                # Configure the tenant network underlay endpoint on ens3np0
                name: ens3np0
                use_dhcp: false
                addresses:
                - ip_netmask: {get_param: TenantIpSubnet}
              - type: interface
                # Don't DHCP on the ens3np1 interface
                name: ens3np1
                use_dhcp: false

outputs:
  OS::stack_id:
    description: The OsNetConfigImpl resource.
    value: {get_resource: OsNetConfigImpl}
```

Note: The PF interface has to be enabled (but no IP address configured) in order for traffic to pass to the representors.

5.5. Modifying the Environment

This section lists the required modifications to the environment file in order to set the required configuration options in the overcloud.

5.5.1. `firstboot.yaml` Template to Set Firmware

Add the template for `firstboot.yaml` to the resource registry:

```
resource_registry:
  # Note: this template will override all other NodeUserData hooks.
  # Switch NFP firmware on SR-IOV nodes
  OS::TripleO::NodeUserData:
    /path/to/firstboot.yaml
```

5.5.2. Network Interface Template for Compute SR-IOV Role

Add the template for the NIC config to the resource registry:

```
resource_registry:
  # Network Interface template for ComputeSriov
  OS::TripleO::ComputeSriov::Net::SoftwareConfig:
    /path/to/nic-configs/compute-sriov.yaml
```

5.5.3. Add the Parameters for the Compute SR-IOV Nodes

The following section illustrates how to configure the Compute SR-IOV nodes to:

- Set the kernel parameters required for IOMMU passthrough,
- Create `32` virtual functions on `ens3np0`, and
- Add those virtual functions to the PCI whitelist.

```
parameter_defaults:
  ComputeSriovParameters:
    KernelArgs: "intel_iommu=on"
    # Enable 32 Virtual Functions on ens3np0
    NeutronSriovNumVFs: ["ens3np0:32"]
    # Add those Virtual Functions to the PCI Passthrough Whitelist
    NovaPCIPassthrough:
        - devname: "ens3np0"
          physical_network: null
          vendor_id: "19ee"
          product_id: "6003"
```

Important Notes:

- Some SmartNIC vendors require the parameter "`:switchdev`" to be appended to the `NeutronSriovNumVFs` configuration. This is not necessary for Netronome SmartNICs.
- The PCI Vendor and Product ID tuple for Netronome Agilio SmartNIC VFs is `19ee:6003`. This is currently irrespective of the model, for more information consult the user guide.

5.5.4. Example Parameters for Neutron Configuration

The following section illustrates how to configure Neutron to:

- Enable vxlan overlays,
- Enable the openvswitch mechanism driver,
- Enable the L2Population and ARP Responder, and
- Configure Nova and Neutron to use the No-op Firewall Driver for stateless forwarding.

```
parameter_defaults:
  NeutronTunnelTypes: 'vxlan'
  NeutronMechanismDrivers: ['openvswitch', 'l2population']
  NeutronEnableL2Pop: True
  NeutronEnableARPResponder: True
  NeutronNetworkType: ['vxlan','vlan','flat']
  ExtraConfig:
    # Extra Configuration to disengage the iptables firewall.
    neutron::plugins::ml2::firewall_driver: neutron.agent.firewall.NoopFirewallDriver
    neutron::agents::ml2::ovs::firewall_driver: neutron.agent.firewall.NoopFirewallDriver
```

```
    # Extra Configuration for Nova
    nova::network::neutron::security_group_api: unset
    nova::network::neutron::firewall_driver: nova.virt.firewall.NoopFirewallDriver
```

5.5.5. Combined Example for offload-environment.yaml

The following snippet combines the previous sections into a single environment template:

```
resource_registry:
  # Note: this template will override all other NodeUserData hooks.
  # Switch NFP firmware on SR-IOV nodes
  OS::TripleO::NodeUserData:
    /path/to/firstboot.yaml
  # Network Interface template for ComputeSriov
  OS::TripleO::ComputeSriov::Net::SoftwareConfig:
    /path/to/nic-configs/compute-sriov.yaml

parameter_defaults:
  NeutronTunnelTypes: 'vxlan'
  NeutronMechanismDrivers: ['openvswitch', 'l2population']
  NeutronEnableL2Pop: True
  NeutronEnableARPResponder: True
  NeutronNetworkType: ['vxlan','vlan','flat']
  ComputeSriovParameters:
    KernelArgs: "intel_iommu=on"
    # Enable 32 Virtual Functions on ens3np0
    NeutronSriovNumVFs: ["ens3np0:32"]
    # Add those Virtual Functions to the PCI Passthrough Whitelist
    NovaPCIPassthrough:
        - devname: "ens3np0"
          physical_network: null
          vendor_id: "19ee"
          product_id: "6003"
  ExtraConfig:
    # Extra Configuration to disengage the iptables firewall.
    neutron::plugins::ml2::firewall_driver: neutron.agent.firewall.NoopFirewallDriver
    neutron::agents::ml2::ovs::firewall_driver: neutron.agent.firewall.NoopFirewallDriver
    # Extra Configuration for Nova
    nova::network::neutron::security_group_api: unset
    nova::network::neutron::firewall_driver: nova.virt.firewall.NoopFirewallDriver
```

5.6. Deploy the Overcloud

The following command illustrates an example deployment using the Director CLI utilities:

```
openstack overcloud deploy \
--templates /usr/share/openstack-tripleo-heat-templates \
-r roles_data.yaml \
-e node_info.yaml \
-e docker_registry.yaml \
-e /usr/share/openstack-tripleo-heat-templates/environments/neutron-ovs-hw-offload.yaml \
-e /usr/share/openstack-tripleo-heat-templates/environments/host-config-and-reboot.yaml \
-e /usr/share/openstack-tripleo-heat-templates/environments/network-isolation.yaml \
-e network-environment.yaml \
-e offload-environment.yaml \
--libvirt-type kvm \
--ntp-server pool.ntp.org
```

Where:

`roles_data.yaml`: The custom roles defined for the deployment
`node_info.yaml`: Number of nodes and mapping to baremetal hints
`docker_registry.yaml`: Container registry information
`neutron-ovs-hw-offload.yaml`: Enables hardware offloads on `ComputeSRIOV` nodes
`network-isolation.yaml`: Standard template enabling network isolation
`network-environment.yaml`: Template with specific network parameters for this deployment
`offload-environment.yaml`: Template with offload configuration for this deployment

5.7. Example Operations

Note: In this section, it is assumed that that the initial setup of the overcloud on the Director has been completed. All commands are assumed to be run as the `stack` user with the admin credentials of the overcloud (`overcloudrc`) sourced:

```
[stack@director ~]$ source overcloudrc
(overcloud) [stack@director ~]$
```

This section details the procedure to launch an instance with an accelerated port.

5.7.1. Uploading Image and Keys

For this example, a RHEL 7.5 server cloud image is uploaded, a flavor capable of hosting it is created, and an administrative key pair is created. The following commands perform those operations:

```
openstack image create "rhel-server-7.5" \
  --file rhel-server-7.5-x86_64-kvm.qcow2 \
  --disk-format qcow2 \
  --container-format bare \
  --public
```

```
openstack flavor create --vcpus 2 --ram 2048 --disk 20 m1.standard

openstack keypair create --public-key ~/.ssh/id_rsa.pub adminkey
```

5.7.2. Creating an Overlay Network

The following commands create an external shared network on the `datacentre` provider network, an internal self-service network and adds a virtual router between them. Note that this configuration depends heavily on the deployment topology. In this example, the external network is accessible via VLAN `112` and has a routable subnet defined on `10.0.112.0/24`. It is assumed that the Director node has been configured with routes to this network.

```
openstack network create --external --share \
    --provider-physical-network datacentre \
    --provider-network-type vlan \
    --provider-segment 112 external
openstack subnet create \
    --allocation-pool start=10.0.112.101,end=10.0.112.200 \
    --network external \
    --dns-nameserver 8.8.8.8 \
    --gateway 10.0.112.1 \
    --no-dhcp \
    --subnet-range 10.0.112.0/24 \
    external-subnet

openstack network create selfservice
openstack subnet create --network selfservice \
  --dns-nameserver 8.8.8.8 \
  --gateway 192.168.42.1 \
  --subnet-range 192.168.42.0/24 \
  selfservice-subnet

openstack router create router
openstack router add subnet router selfservice-subnet
openstack router set --external-gateway external router
```

5.7.3. Creating a Port and Launching an Instance

The following commands create a port on the self-service network with the required offload capabilities set. Note that the "`switchdev`" capability is required in order for the OVS mechanism driver to recognise that the port needs to be offloaded. An instance is launched and a floating IP is assigned to it.

```
openstack port create --network selfservice \
```

```
    --vnic-type direct \
    --binding-profile '{"capabilities": ["switchdev"]}' \
    port-vnic-direct

FLOATING_IP_JSON=$(openstack floating ip create external -f json)
FLOATING_IP=$(echo $FLOATING_IP_JSON | jq -r '.floating_ip_address')

openstack server create --flavor m1.standard \
    --image rhel-server-7.5 \
    --nic port-id=port-vnic-direct \
    --key adminkey \
    instance-direct

openstack server add floating ip instance-direct $FLOATING_IP
```

5.7.3. Connecting to the Instance

At this point, the instance should be accessible on the floating IP via the provider network. To verify that the instance has the correct PCI device attached, connect via SSH and use ethtool to query the driver:

```
(overcloud) [stack@undercloud ~]$ ssh cloud-user@$FLOATING_IP
Last login: Wed May 23 06:30:51 2018 from 10.0.112.3
[cloud-user@instance-direct ~]$ ethtool -i eth0
driver: nfp
version: 3.10.0-862.el7.x86_64 SMP mod_u
firmware-version: 0.0.3.0
expansion-rom-version:
bus-info: 0000:00:04.0
supports-statistics: yes
supports-test: no
supports-eeprom-access: no
supports-register-dump: yes
supports-priv-flags: no
```

Appendix A: unabridged `nic-configs/compute-sriov.yaml`

```
heat_template_version: queens

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
  BondInterfaceOvsOptions:
    default: 'bond_mode=active-backup'
    description: The ovs_options string for the bond interface. Set things like
                 lacp=active and/or bond_mode=balance-slb using this option.
    type: string
  ExternalNetworkVlanID:
    default: 10
    description: Vlan ID for the external network traffic.
    type: number
  InternalApiNetworkVlanID:
    default: 20
    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: 50
    description: Vlan ID for the tenant network traffic.
    type: number
  ManagementNetworkVlanID:
    default: 60
    description: Vlan ID for the management network traffic.
    type: number
  ExternalInterfaceDefaultRoute:
    default: '10.0.0.1'
    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: ovs_bridge
                name: bridge_name
                use_dhcp: false
                dns_servers: {get_param: DnsServers}
                addresses:
                - ip_netmask:
                    list_join:
                    - '/'
                    - - {get_param: ControlPlaneIp}
                      - {get_param: ControlPlaneSubnetCidr}
                routes:
                - default: true
                  next_hop: {get_param: ControlPlaneDefaultRoute}
                - ip_netmask: 169.254.169.254/32
                  next_hop: {get_param: EC2MetadataIp}
                members:
```

```
                - type: interface
                  name: eno1
                  # force the MAC address of the bridge to this interface
                  primary: true
                - type: vlan
                  vlan_id: {get_param: InternalApiNetworkVlanID}
                  addresses:
                  - ip_netmask: {get_param: InternalApiIpSubnet}
              - type: ovs_bridge
                name: br-storage
                use_dhcp: false
                members:
                - type: interface
                  name: eno3
                  # force the MAC address of the bridge to this interface
                  primary: true
                - type: vlan
                  vlan_id: {get_param: StorageNetworkVlanID}
                  addresses:
                  - ip_netmask: {get_param: StorageIpSubnet}
              - type: interface
                # Enable the PF interface but don't DHCP on it
                name: ens3
                use_dhcp: false
              - type: interface
                name: ens3np0
                use_dhcp: false
                addresses:
                - ip_netmask: {get_param: TenantIpSubnet}
              - type: interface
                # Don't DHCP on the ens3np1 interface
                name: ens3np1
                use_dhcp: false
              - type: interface
                # Don't DHCP on the eno2 interface
                name: eno2
                use_dhcp: false
              - type: interface
                # Don't DHCP on the eno4 interface
                name: eno4
                use_dhcp: false

outputs:
  OS::stack_id:
    description: The OsNetConfigImpl resource.
    value: {get_resource: OsNetConfigImpl}
```

Appendix B: Offloadable Flows

Flows may be offloaded to hardware if they meet the criteria described in this section.

B.1. Matches

A flow may be offloaded if it matches only on the following fields:

Metadata	Input Port
Layer 2	Ethernet: Type, Addresses
	VLAN: Outermost ID, Priority
Layer 2.5	MPLS: Outermost Label, TC, BoS
Layer 3	IPv4: Addresses, Protocol, TTL, TOS, Frag
	IPv6: Addresses, Protocol, Hop Limit, TOS, Frag
Layer 4	TCP: Ports, Flags
	UDP: Ports
	SCTP: Ports
Tunnel	ID
	IPv4: Outer Address
	IPv6: Outer Address
	UDP: Outer Destination Port

B.2. Actions

A flow may be offloaded if:

- The flow has no actions (drop) or outputs to one port
- The in port of the flow is:
    - A physical port or VF on an Agilio SmartNIC or;
    - A VXLAN vport whose ingress packets are received on a physical port or VF on an Agilio SmartNIC (flow rule must include outer IP addresses and well known VXLAN port 4789) and whose egress action is to a physical port or VF on an Agilio SmartNIC.
- If present, the output action outputs to:
    - A physical port or VF on the same Agilio SmartNIC as the in port
    - A VXLAN vport whose egress packets are sent on a physical port or VF of the Agilio SmartNIC as the in port.
- Only the in port our output port may be a VXLAN tunnel, not both

Other than output and the implicit drop action, flows using the following actions may be offloaded:

- Push and Pop VLAN
- Masked and Unmasked Set

Flows that include a masked set of any of the following fields may be offloaded:

Layer 2	Ethernet: Type, Addresses
	VLAN: ID, Priority
Layer 3	IPv4: Addresses, TTL
	IPv6: Addresses, Hop Limit
Layer 4	TCP: Ports
	UDP: Ports
	SCTP: Ports

Flows that include an unmasked set of any of the following fields may be offloaded:

Tunnel	ID
	IPv4: Outer Address
	IPv6: Outer Address
	UDP: Outer Destination Port

B.3. Bonds

Flows resulting from the following modes could be accelerated:

OVS Bonds Modes	Active Backup
	Balance SLB
	Balance TCP

Configuring a bond in OVS in active-backup or balance-slb modes would result in flows that are offloadable. It should be noted that OVS sends packets to the LOCAL port for each bond. This results in flow rules that include actions with output to the LOCAL port. This cannot be accelerated by Agilio OVS. To prevent this from occurring, and to achieve acceleration, packets must not be sent to the LOCAL port, which can be achieved by:

```
ovs-ofctl -OOpenflow13 mod-port bondbr0 bondbr0 no-forward
```

Furthermore configuring a bond in balance-tcp mode could result in flows that are offloadable if recirculation has been disabled. This can be achieved using the following:

```
ovs-appctl dpif/set-dp-features bondbr0 recirc false
```

It should be noted that turning off recirculation leads to exact match datapath entries (matching on L2, L3 and L4) being installed. e.g.

```
in_port(10),eth(src=12:23:34:45:56:67,dst=67:56:45:34:23:12),eth_type(0x0800),ipv4(src=10
.10.10.10,dst=10.10.10.20,proto=6,frag=no),tcp(src=1000,dst=2000), packets:0, bytes:0,
used:never, actions:6,7
```

This exact matching behaviour leads to flow explosion, i.e. OVS will install an entry for every unique (L2, L3 or L4) packet. This in turn could lead to performance degradation, especially so when using many flows (100K and more).

Finally, OVS bonding is based on the NORMAL rule; links will not be aggregated when the bond bridge does not contain a NORMAL rule. Should match/actions be required, an additional bridge (named br0 in this example) is required on which the match/actions are performed, allowing the bond bridge to only have the NORMAL rule. This additional bridge can be connected to the bond bridge using a patch port.

Contact Us

Netronome Systems, Inc.
2903 Bunker Hill Lane, Suite 150
Santa Clara, CA 95054
Tel: 408.496.0022 | Fax: 408.586.0002
www.netronome.com | support@netronome.com