Red Hat OpenStack Platform 13

Network Functions Virtualization Product Guide

Overview of the Network Functions Virtualization (NFV)

Last Updated: 2024-03-21

Red Hat OpenStack Platform 13 Network Functions Virtualization Product Guide

Overview of the Network Functions Virtualization (NFV)

OpenStack Team
rhos-docs@redhat.com

Legal Notice

Abstract

This guide introduces Network Functions Virtualization (NFV), its advantages, supported configurations, architecture, components, installation, and integration information.

Table of Contents

MAKING OPEN SOURCE MORE INCLUSIVE

Red Hat is committed to replacing problematic language in our code, documentation, and web properties. We are beginning with these four terms: master, slave, blacklist, and whitelist. Because of the enormity of this endeavor, these changes will be implemented gradually over several upcoming releases. For more details, see our CTO Chris Wright’s message .

CHAPTER 1. UNDERSTANDING RED HAT NETWORK FUNCTIONS VIRTUALIZATION (NFV)

Network Functions Virtualization (NFV) is a software-based solution that helps the Communication Service Providers (CSPs) to move beyond the traditional, proprietary hardware to achieve greater efficiency and agility while reducing the operational costs.

An NFV environment allows for IT and network convergence by providing a virtualized infrastructure using the standard virtualization technologies that run on standard hardware devices such as switches, routers, and storage to virtualize network functions. The management and orchestration logic deploys and sustains these services. NFV also includes Systems Administration, Automation, and Life-Cycle Management thereby reducing the manual work necessary.

1.1. ADVANTAGES OF NFV

The main advantages of implementing network functions virtualization (NFV) are as follows:

- Accelerates the time-to-market by allowing you to to quickly deploy and scale new networking services to address changing demands.
- Supports innovation by enabling service developers to self-manage their resources and prototype using the same platform that will be used in production.
- Addresses customer demands in hours or minutes instead of weeks or days, without sacrificing security or performance.
- Reduces capital expenditure because it uses commodity-off-the-shelf hardware instead of expensive tailor-made equipment.
- Uses streamlined operations and automation that optimize day-to-day tasks to improve employee productivity and reduce operational costs.

1.2. SUPPORTED CONFIGURATIONS FOR NFV DEPLOYMENTS

Red Hat OpenStack Platform supports NFV deployments with the inclusion of automated OVS-DPDK and SR-IOV configuration.

Hyper-converged Infrastructure (HCI)

Customers can now co-locate the Compute sub-system with the Red Hat Ceph Storage nodes. This hyper-converged model delivers lower cost of entry, smaller initial deployment footprints, maximized capacity utilization, and more efficient management in NFV use cases.

Composable roles

You can now use composable roles to create custom deployments. Composable roles allow you to add or remove services from each role. For more information on the Composable Roles, see Composable Roles and Services.

Open vSwitch (OVS) with LACP

As of OVS 2.9, LACP with OVS is fully supported. This is not recommended for Openstack control plane traffic, as OVS or Openstack Networking interruptions may interfere with management. For more information See Open vSwitch Bonding Options.

OVS Hardware offload (Technology Preview)

Red Hat OpenStack Platform supports, with limitations, the deployment of OVS hardware offload as a technology preview. For information about deploying OVS with hardware offload, see OpenvSwitch Hardware offload .
For more information on the support scope for features marked as technology previews, see Technology Preview.

NOTE

Red Hat does not support the use of Open Virtual Network (OVN) with Network Functions Virtualization infrastructure (NFVi).

CHAPTER 2. NFV SOFTWARE

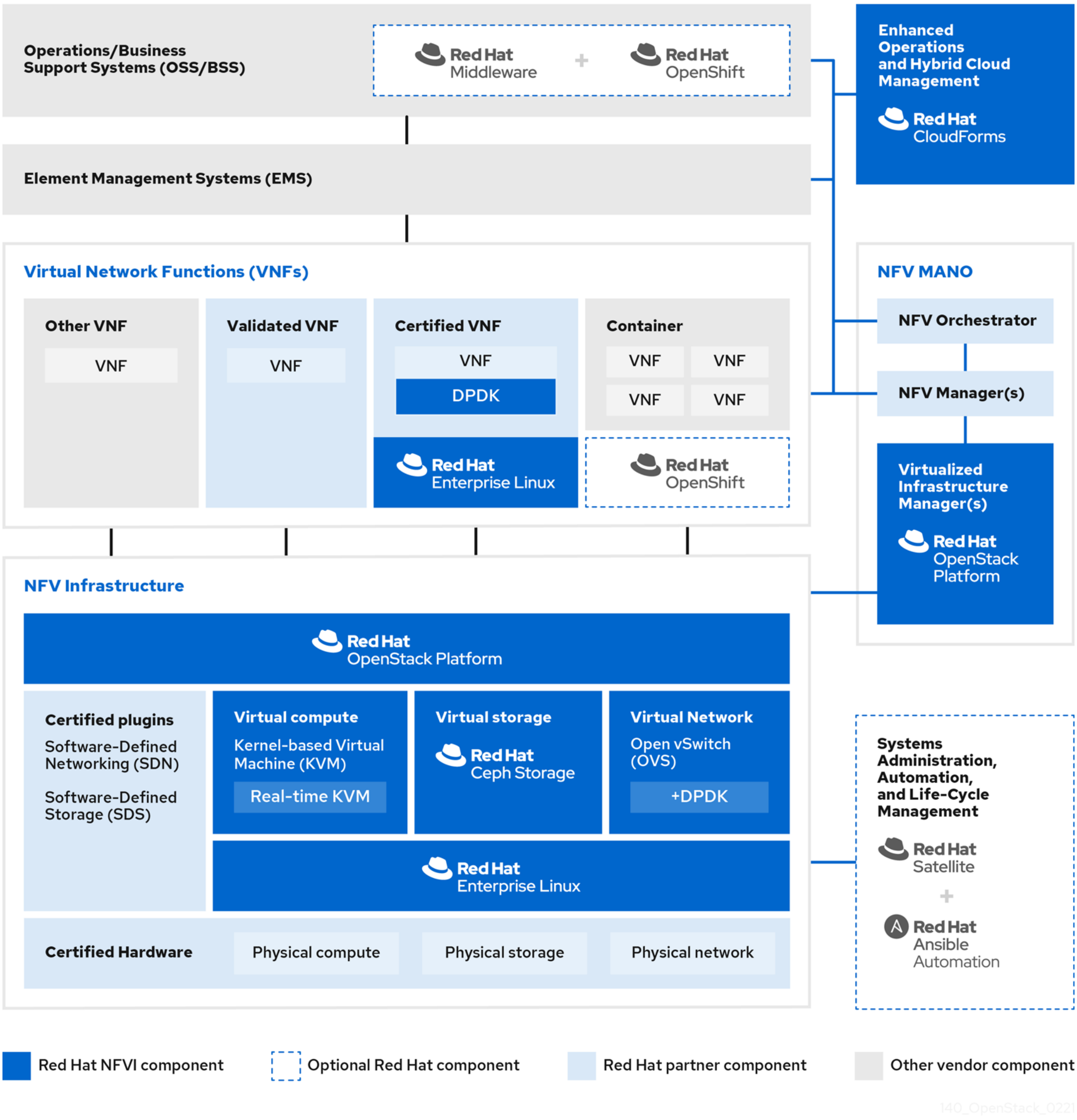
2.1. ETSI NFV ARCHITECTURE

The European Telecommunications Standards Institute (ETSI) is an independent standardization group that develops standards for information and communications technologies (ICT) in Europe.

Network functions virtualization (NFV) focuses on addressing problems involved in using proprietary hardware devices. NFV reduces the need to install network-specific equipment, depending upon the use case requirements and economic benefits. The ETSI Industry Specification Group for Network Functions Virtualization (ETSI ISG NFV) sets the requirements, reference architecture, and the infrastructure specifications necessary to ensure virtualized functions are supported.

Red Hat is offering an open-source based cloud-optimized solution to help the Communication Service Providers (CSP) to achieve IT and network convergence. Red Hat adds NFV features such as single root I/O virtualization (SR-IOV) and Open vSwitch with Data Plane Development Kit (OVS-DPDK) to Red Hat OpenStack.

2.2. NFV ETSI ARCHITECTURE AND COMPONENTS



In general, a network functions virtualization (NFV) platform has the following components:

- **Virtualized Network Functions (VNFs)** - the software implementation of network functions, such as routers, firewalls, load balancers, broadband gateways, mobile packet processors, servicing nodes, signalling, location services, and other network functions.

- **NFV Infrastructure (NFVi)** - the physical resources (compute, storage, network) and the virtualization layer that make up the infrastructure. The network includes the datapath for forwarding packets between virtual machines and across hosts. This allows you to install VNFs without being concerned about the details of the underlying hardware. NFVi forms the foundation of the NFV stack. NFVi supports multi-tenancy and is managed by the Virtual Infrastructure Manager (VIM). Enhanced Platform Awareness (EPA) improves the virtual machine packet forwarding performance (throughput, latency, jitter) by exposing low-level CPU and NIC acceleration components to the VNF.

- **NFV Management and Orchestration (MANO)**- the management and orchestration layer focuses on all the service management tasks required throughout the lifecycle of the VNF. The main goals of MANO are to allow service definition, automation, error-correlation, monitoring, and life-cycle management of the network functions offered by the operator to its customers, decoupled from the physical infrastructure. This decoupling requires additional layers of management, provided by the Virtual Network Function Manager (VNFM). VNFM manages the life cycle of the virtual machines and VNFs, either by interacting with them directly, or through the Element Management System (EMS) provided by the VNF vendor. The other important component defined by MANO is the Orchestrator also known as NFVO. NFVO interfaces with various databases and systems including Operations/Business Support Systems (OSS/BSS) on the top and with the VNFM on the bottom. If the NFVO wants to create a new service for a customer, it asks the VNFM to trigger the instantiation of a VNF (which may result in multiple virtual machines).

- **Operations and Business Support Systems (OSS/BSS)**- provides the essential business function applications, for example, operations support and billing. The OSS/BSS needs to be adapted to NFV, integrating with both legacy systems and the new MANO components. The BSS systems set policies based on service subscriptions and manage reporting and billing.

- **Systems Administration, Automation and Life-Cycle Management**- manages system administration, automation of the infrastructure components and life-cycle of the NFVi platform.

2.3. RED HAT NFV COMPONENTS

Red Hat’s solution for NFV includes a range of products that can act as the different components of the NFV framework in the ETSI model. The following products from the Red Hat portfolio integrate into an NFV solution:

- Red Hat OpenStack Platform - Supports IT and NFV workloads. The Enhanced Platform Awareness (EPA) features deliver deterministic performance improvements through CPU Pinning, Huge pages, Non-Uniform Memory Access (NUMA) affinity, and network adaptors (NICs) that support SR-IOV and OVS-DPDK.

- Red Hat Enterprise Linux and Red Hat Enterprise Linux Atomic Host - Create virtual machines and containers as VNFs.

- Red Hat Ceph Storage - Provides the unified elastic and high-performance storage layer for all the needs of the service provider workloads.

- Red Hat JBoss Middleware and OpenShift Enterprise by Red Hat - Optionally provide the ability to modernize the OSS/BSS components.

- Red Hat CloudForms - Provides a VNF manager and presents data from multiple sources, such as the VIM and the NFVi, in a unified display.

- Red Hat Satellite and Ansible by Red Hat - Optionally provide enhanced systems administration, automation and life-cycle management.

2.4. NFV INSTALLATION SUMMARY

The Red Hat OpenStack Platform director installs and manages a complete OpenStack environment. The director is based on the upstream OpenStack TripleO project, which is an abbreviation for "OpenStack-On-OpenStack". This project takes advantage of the OpenStack components to install a fully operational OpenStack environment; this includes a minimal OpenStack node called the undercloud. The undercloud provisions and controls the overcloud (a series of bare metal systems used as the production OpenStack nodes). The director provides a simple method for installing a complete Red Hat OpenStack Platform environment that is both lean and robust.

For more information on installing the undercloud and overcloud, see Red Hat OpenStack Platform Director Installation and Usage.

For more information about NFV planning guidelines and configuration, see Network Function Virtualization Planning and Configuration Guide.

CHAPTER 3. NFV HARDWARE

See Director Installation and Usage for guidance on hardware selection for OpenStack nodes.

For a list of tested NICs for network functions virtualization (NFV), see Network Adapter Support. (Customer Portal login required.)

CHAPTER 4. NFV DATA PLANE CONNECTIVITY

With the introduction of NFV, more networking vendors are starting to implement their traditional devices as VNFs. While the majority of networking vendors are considering virtual machines, some are also investigating a container-based approach as a design choice. An OpenStack-based solution should be rich and flexible due to two primary reasons:

- **Application readiness** - Network vendors are currently in the process of transforming their devices into VNFs. So different VNFs in the market have different maturity levels; common barriers to this readiness include enabling RESTful interfaces in their APIs, evolving their data models to become stateless, and providing automated management operations. OpenStack should provide a common platform for all.

- **Broad use-cases** - NFV includes a broad range of applications that serve different use-cases. For example, Virtual Customer Premise Equipment (vCPE) aims at providing a number of network functions such as routing, firewall, VPN, and NAT at customer premises. Virtual Evolved Packet Core (vEPC), is a cloud architecture that provides a cost-effective platform for the core components of Long-Term Evolution (LTE) networks, allowing dynamic provisioning of gateways and mobile endpoints to sustain the increased volumes of data traffic from smartphones and other devices.
  These use-cases, by nature, are implemented using different network applications and protocols, and require different connectivity, isolation and performance characteristics from the infrastructure. It is also common to separate between control plane interfaces and protocols and the actual forwarding plane. It is necessary for OpenStack to be flexible enough to offer different datapath connectivity options.

In principle, there are two common approaches for providing data plane connectivity to virtual machines:

- **Direct hardware access**bypasses the linux kernel and provides secure direct memory access (DMA) to the physical NIC using technologies such as PCI Passthrough or single root I/O virtualization (SR-IOV) for both Virtual Function (VF) and Physical Function (PF) pass-through.

- **Using a virtual switch (vswitch)**, implemented as a software service of the hypervisor. Virtual machines are connected to the vSwitch using virtual interfaces (vNICs), and the vSwitch is capable of forwarding traffic between virtual machines as well as between virtual machines and the physical network.

4.1. FAST DATA PATH OPTIONS

Some of the fast data path options are as follows:

- **Single Root I/O Virtualization (SR-IOV)**- is a standard that makes a single PCI hardware device appear as multiple virtual PCI devices. It works by introducing Physical Functions (PFs), which are the fully featured PCIe functions that represent the physical hardware ports, and Virtual Functions (VFs), which are lightweight functions that are assigned to the virtual machines. For the VM, the VF resembles a regular NIC that communicates directly with the hardware. NICs support multiple VFs.

- **Open vSwitch (OVS)** - is an open source software switch that is designed to be used as a virtual switch within a virtualized server environment. OVS supports the capabilities of a regular L2-L3 switch and also offers support to the SDN protocols such as OpenFlow to create user-defined overlay networks (for example, VXLAN). OVS uses Linux kernel networking to switch packets between virtual machines and across hosts using physical NIC. OVS now supports connection tracking (Conntrack) with built-in firewall capability to avoid the overhead of Linux bridges that use iptables/ebtables. Open vSwitch for Red Hat OpenStack Platform environments offers default OpenStack Networking (neutron) integration with OVS.

- **Data Plane Development Kit (DPDK)** - consists of a set of libraries and poll mode drivers (PMD) for fast packet processing. It is designed to run mostly in the user-space, enabling applications to perform their own packet processing directly from/to the NIC. DPDK reduces latency and allows more packets to be processed. DPDK Poll Mode Drivers (PMDs) run in busy loop, constantly scanning the NIC ports on host and vNIC ports in guest for arrival of packets.

- **DPDK accelerated Open vSwitch (OVS-DPDK)** - is Open vSwitch bundled with DPDK for high performance user-space solution with Linux kernel bypass and direct memory access (DMA) to physical NICs. The idea is to replace the standard OVS kernel data path with a DPDK-based data path, creating a user-space vSwitch on the host, which uses DPDK internally for its packet forwarding. The advantage of this architecture is that it is mostly transparent to users as the basic OVS features as well as the interfaces it exposes (such as OpenFlow, OVSDB, the command line) remain mostly the same.

CHAPTER 5. NFV PERFORMANCE CONSIDERATIONS

For a network functions virtualization (NFV) solution to be useful, its virtualized functions must meet or exceed the performance of physical implementations. Red Hat's virtualization technologies are based on the high-performance Kernel-based Virtual Machine (KVM) hypervisor, common in OpenStack and cloud deployments.

5.1. CPUS AND NUMA NODES

Previously, all memory on x86 systems was equally accessible to all CPUs in the system. This resulted in memory access times that were the same regardless of which CPU in the system was performing the operation and was referred to as Uniform Memory Access (UMA).

In Non-Uniform Memory Access (NUMA), system memory is divided into zones called nodes, which are allocated to particular CPUs or sockets. Access to memory that is local to a CPU is faster than memory connected to remote CPUs on that system. Normally, each socket on a NUMA system has a local memory node whose contents can be accessed faster than the memory in the node local to another CPU or the memory on a bus shared by all CPUs.

Similarly, physical NICs are placed in PCI slots on the Compute node hardware. These slots connect to specific CPU sockets which are associated to a particular NUMA node. For optimum performance, connect your datapath NICs to the same NUMA nodes in your CPU configuration (SR-IOV or OVS-DPDK).

The performance impact of NUMA misses are significant, generally starting at a 10% performance hit or higher. Each CPU socket can have multiple CPU cores which are treated as individual CPUs for virtualization purposes.

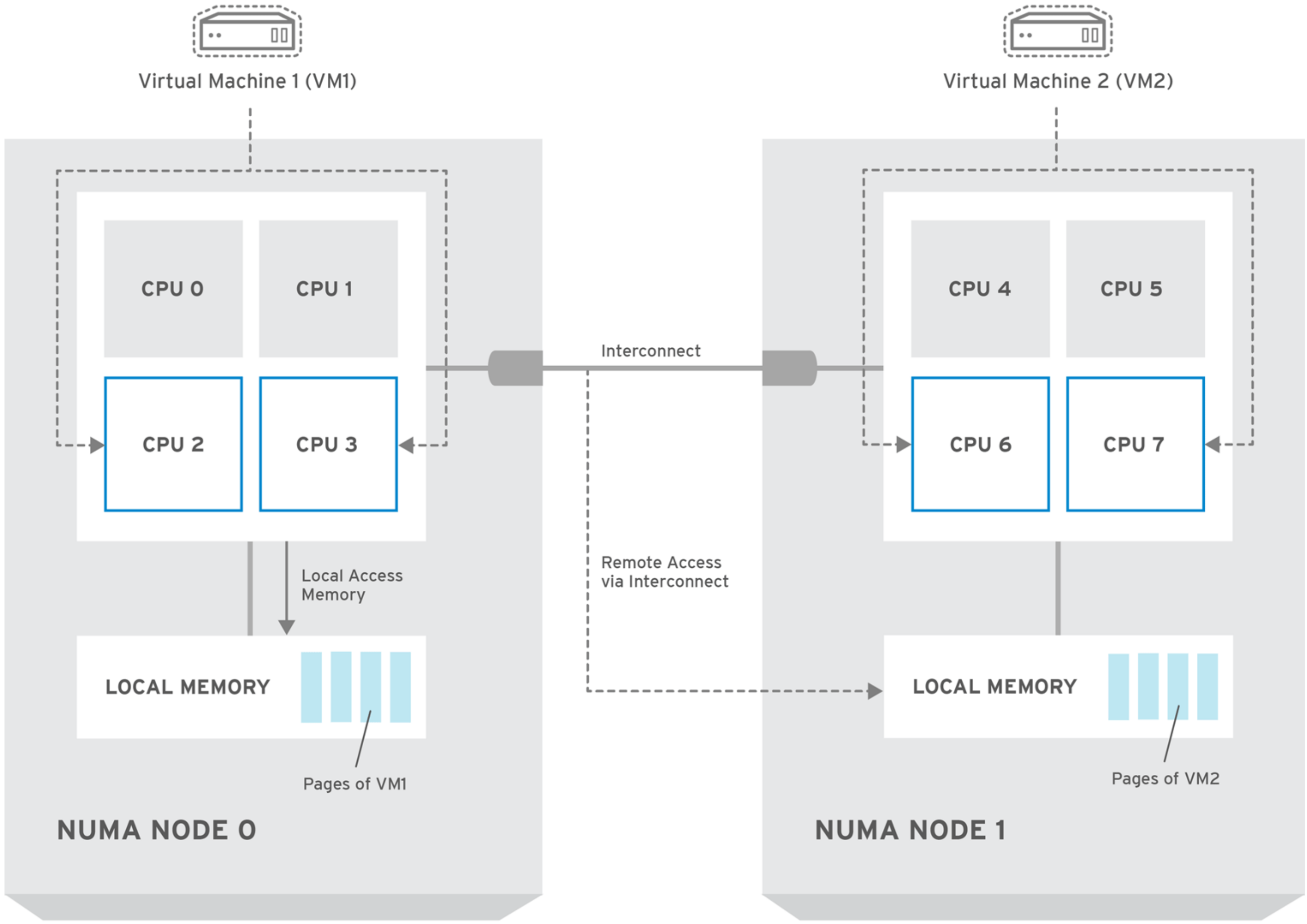
OpenStack Compute makes smart scheduling and placement decisions when launching instances. Administrators who want to take advantage of these features can create customized performance flavors to target specialized workloads including NFV and High Performance Computing (HPC).

TIP

Background information about NUMA is available in the following article: What is NUMA and how does it work on Linux?

5.2. NUMA NODE EXAMPLE

The following diagram provides an example of a two-node NUMA system and the way the CPU cores and memory pages are made available:



OPENSTACK_ 39825_0516

NOTE

Remote memory available via Interconnect is accessed **only** if VM1 from NUMA node 0 has a CPU core in NUMA node 1. In this case, the memory of NUMA node 1 will act as local for the third CPU core of VM1 (for example, if VM1 is allocated with CPU 4 in the diagram above), but at the same time, it will act as remote memory for the other CPU cores of the same VM.

5.3. CPU PINNING

CPU pinning is the ability to run a specific virtual machine's virtual CPU on a specific physical CPU, in a given host. vCPU pinning provides similar advantages to task pinning on bare-metal systems. Since virtual machines run as user space tasks on the host operating system, pinning increases cache efficiency.

See Configure CPU Pinning with NUMA for further information.

5.4. HUGEPAGES

Physical memory is segmented into contiguous regions called pages. For efficiency, the system retrieves memory by accessing entire pages instead of individual bytes of memory. To perform this translation, the system looks in the Translation Lookaside Buffers (TLB) which contain the physical to virtual address mappings for the most recently or frequently used pages. When the system cannot find a mapping in the TLB, the processor must iterate through all of the page tables to determine the address mappings. Optimize the TLB to minimize the performance penalty that occurs during these TLB misses.

The typical page size in an x86 system is 4KB. Larger page sizes mean that there are fewer pages overall, and this increases the amount of system memory that can have its virtual to physical address translation stored in the TLB. Consequently, this reduces TLB misses, which increases performance. With larger page sizes, there is an increased potential for memory to be under-utilized as processes must allocate in pages, but not all of the memory is likely required. As a result, choosing a page size is a compromise between providing faster access times with larger pages, and ensuring maximum memory utilization with smaller pages.

5.5. PORT SECURITY

Port security is an anti-spoofing measure that blocks any egress traffic that does not match the source IP and source MAC address of the originating network port. You cannot view or modify this behavior using security group rules.

By default, the **port_security_enabled** parameter is set to **enabled** on newly created Neutron networks in OpenStack. Newly created ports copy the value of the **port_security_enabled** parameter from the network they are created on.

For some NFV use cases, such as building a firewall or router, you must disable port security.

To prevent port security from being enabled on any newly created port on a network, set the port_security_enabled parameter on the network to disabled:

To disable port security on a single port, run the following command:

```
openstack port set --disable-port-security <port-id>
```

To prevent port security from being enabled on any newly created port on a network, run the following command:

```
openstack network set --disable-port-security <network-id>
```

CHAPTER 6. 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 6.1. 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	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*. Ensure that you have 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. 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.
NFV Documentation	For more details on planning and configuring your Red Hat OpenStack Platform deployment with single root I/O virtualization (SR-IOV) and Open vSwitch with Data Plane Development Kit (OVS-DPDK), see Network Function Virtualization Planning and Configuration Guide.