Red Hat OpenStack Platform 8

Red Hat Solution for Network Functions Virtualization

Introduction to the Red Hat Solution for Network Functions Virtualization

OpenStack Team

Red Hat OpenStack Platform 8 Red Hat Solution for Network Functions Virtualization

Introduction to the Red Hat Solution for Network Functions Virtualization

OpenStack Team
rhos-docs@redhat.com

Legal Notice

Copyright © 2017 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 article introduces Red Hat Solution for NFV and a brief overview of the features in different Red Hat products that can serve as layers for the European Telecommunication Standards Institute (ETSI) NFV model.

Table of Contents

1. OVERVIEW

Note

This article describes the Red Hat solution for Network Function Virtualization which is currently provided as a Technology Preview. For more information on the support scope for features marked as technology previews, see https://access.redhat.com/support/offerings/techpreview/.

Network Functions Virtualization (NFV) uses the concept of virtualization to move entire classes of network node functions into building blocks that interconnect to create communication services. NFV is a new way to define, create, and manage networks by replacing dedicated hardware appliances with software and automation.

A **Virtualized Network Function (VNF)** consists of virtual execution environments running software on standard high-volume servers, switches and storage, instead of using custom hardware appliances for each network function. While the majority of networking vendors are looking into virtual machines (VMs) as execution environments, some are also looking at a container-based approach, hoping to offer more efficiency and agility. As service providers move to being more application-focused and increase the benefits of virtualization by moving the overhead (cost and installation of hardware resources) into the cloud, containers are becoming increasingly important. Containers can create a package around a piece of software that includes everything it needs to run, including code, runtime, and system tools. The software can then be run the same way in different operating environments. Containers provide the advantages such as increased density, rapid installation time, and so on.

NFV virtualizes network functions on general-purpose, cloud-based infrastructure to provide more agility, flexibility, simplicity, efficiency, and scalability than legacy infrastructure, while also reducing costs and allowing greater innovation.

An NFV environment is designed to consolidate and deliver the networking components needed to support a fully virtualized infrastructure, including virtual servers, storage, and even other networks. It utilizes standard IT virtualization technologies that run on high-volume service, switch, and storage hardware to virtualize network functions. The management and orchestration logic deploys and sustains these services. NFV also has another component - **Network Functions Virtualization Infrastructure (NFVi)** - which is the pool of resources used to host and connect virtual functions. The NFVi resources are managed and controlled by the **Virtualized Infrastructure Manager (VIM)**. These resources are used by the VNFs and the **NFV Management and Orchestration (MANO)** as necessary. One of the main goals of NFV is to decouple the hardware and the software. This decoupling requires additional layer of management. This management is addressed by the NFV MANO. MANO defines the **Virtual Network Functions Manager (VNFM)** which manages the lifecycle of the virtual machines and VNFs by interacting with the virtual machines and VNFs. The other important component defined by MANO is the **Orchestrator**. The Orchestrator interfaces with various databases and systems including **Operation/Business Support Systems (OSS/BSS)** and with the VNF manager. If the Orchestrator wants to create an instance of a VNF or a virtual machine, it requests the VNF manager to create one such instance.

The main business benefits of implementing NFV are as follows:

- It accelerates the time-to-market by allowing quick deployment of new networking services because you do not need to install new hardware to support changing business requirements. NFV lowers the risks associated with rolling out new services, allowing providers to easily trial and evolve services to determine what best meets the needs of customers.

- It delivers agility and flexibility by allowing you to quickly scale services to address changing demands, and supports innovation by enabling services to be delivered using software on any industry-standard server hardware.
- It addresses customer demands in hours or minutes instead of weeks or days, without sacrificing security or performance.
- It reduces capital expenditures because it allows one-time investments.
- It reduces operational costs by reducing the space, power, and cooling requirements of equipment.

1.1. Red Hat and NFV

Red Hat offers an open source-based carrier grade solution: the **Red Hat Solution for NFV** to help Communication Service Providers (CSP) to achieve IT and network convergence, by adding the NFV features to be available on existing open source products like OpenStack.

The most common use cases for the Red Hat NFV Solution fall under broad categories are as follows:

Mobility - Helping Mobile CSP deploy services such as vEPC and vIMS.

Managed Services - Managed Services are services that service providers offer to small, medium, and large enterprises. Most common services are managed Customer Premise Equipment (CPE), managed Firewall, VPN termination gateways, Intrusion Detection and Analytics.

With managed services SPs offer these “aaS” (as a Service) - CPEaaS, FWaaS etc. For managed services, VNFs provides virtualization, but the ability to create service chains to link these services that is more important. This will require not only conventional routing and layer-2 switching within the cloud but also Software-Defined Networking (SDN) to program flows beyond layer-2 and layer-3 based criteria.

Video - Cable operators (MSOs) and other Over The Top (OTT) providers such as Netflix, Amazon Video use a lot of servers and network devices. The general term used for such networks is Content Delivery Network (CDN). This is another area where there are huge operational cost savings from using virtualized cloud instances of devices rather than purpose built hardware. These virtualized CDNs can reside in data centers closer to where the distribution networks are. It is easier for scaling them when they need more capacity, duplicating them as well as for performing disaster recovery.

Note

Red Hat’s approach is to insert the NFV features in many of Red Hat’s products and make them compliant with carrier expectations. We refer to this as a carrier-grade, open-source NFV solution. Red Hat does not intend to create a separate NFV/Telco product.

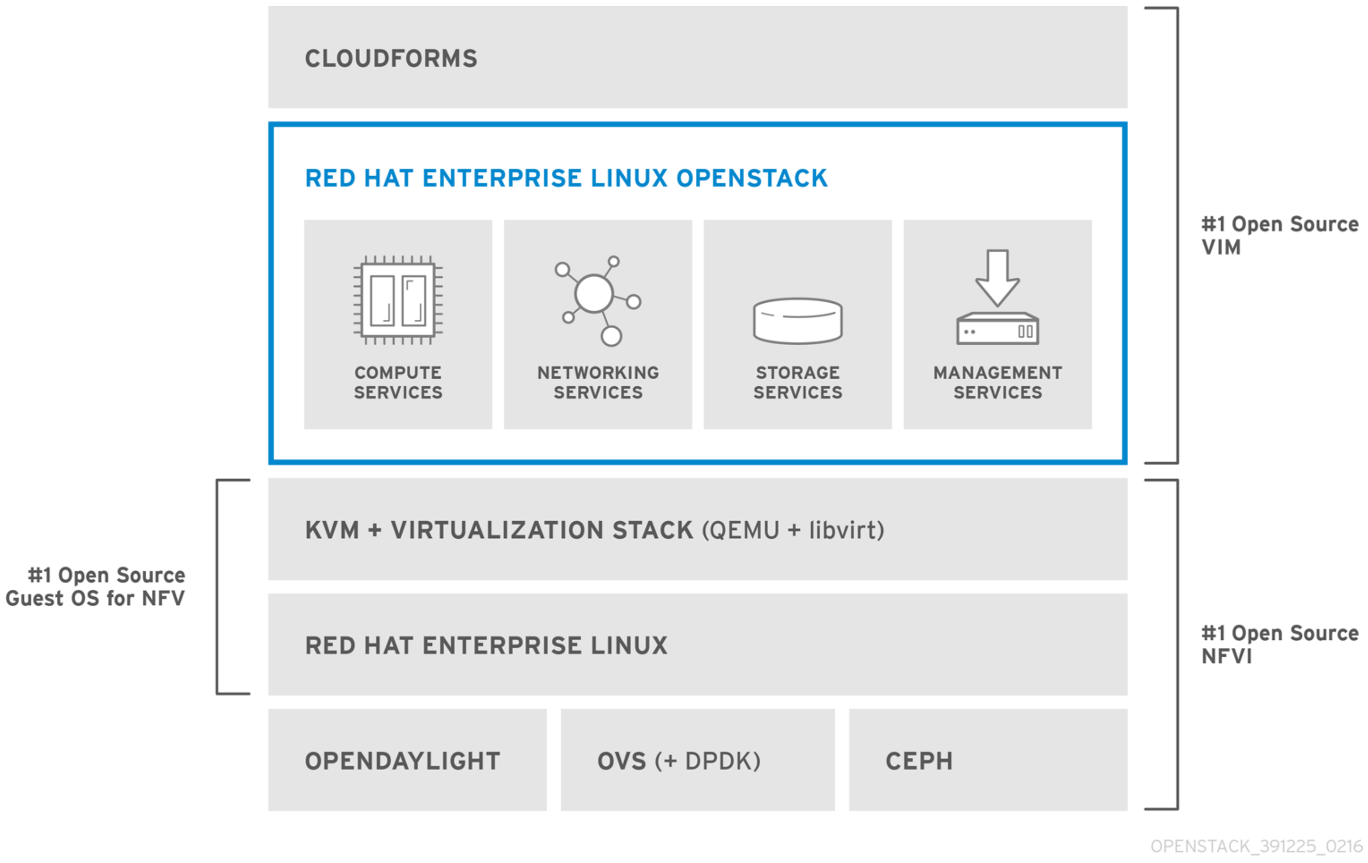
The benefit for service providers in having a common unique platform is IT and network convergence. Red Hat has the following products that can be used for NFV architecture:

- Red Hat OpenStack Platform, a service provider can run IT and NFV workloads as a the VIM.
- Red Hat Enterprise Linux and Red Hat Enterprise Linux Atomic Host allow the creation of virtual machines and containers as VNFs.
- Red Hat JBoss and OpenShift Enterprise can be modified to serve as the OSS/BSS layer.
- Red Hat CloudForms acts as the VNF manager and presents data from multiple sources like the VIM and the NFVi in a unified display.

1.2. Design Environment

The Red Hat NFV Solution consists of several products in the Red Hat portfolio integrating many technologies:

- Red Hat Enterprise Linux (RHEL)
- Red Hat OpenStack Platform with KVM and Open vSwitch
- Red Hat CloudForms
- Red Hat Ceph Storage



The Red Hat NFV Solution consists of the following layers:

- Infrastructure Layer
- Virtualization Layer
- OpenStack Layer
- Management Layer

1.2.1. Infrastructure Layer

The Infrastructure layer is the foundation of the Red Hat NFV Solution. The main components are:

- Red Hat Enterprise Linux (RHEL)
- Software-Defined Storage with Ceph
- Software-Defined Networking (SDN) with OpenDaylight or a certified 3rd party controller

- Open vSwitch (OVS) standalone, or accelerated with Data Plane Development Kit (DPDK)

Key facts:

- The key aspects of RHEL for NFV use cases are High Availability flavor and Real Time flavor.
- RHEL Atomic host is a variation of Red Hat Enterprise Linux 7 optimized to run Linux containers in the Docker format. It uses SELinux to provide strong safeguards in multi-tenant environments, and provides the ability to perform atomic upgrades and rollbacks, enabling quicker and easier maintenance with less downtime. RHEL Atomic is well suited to run VNF as containers
- Ceph provides storage characteristics like performance, scale and reliability
- OpenDayLight is an open source SDN controller. The Red Hat NFV Solution is as well certified with multiple commercial SDN offerings, for example, Nuage, Contrail
- RHEL supports DPDK and Direct Device technologies (SR-IOV, PCI Passthrough). DPDK is available as technology preview in Red Hat OpenStack Platform 8 release as a part of the ‘Extras’ channel.

1.2.2. Virtualization Layer

Virtualization layer is provided by Red Hat Enterprise Virtualization Hypervisor which is mainly composed of:

- Kernel-based Virtual Machine (KVM) and Quick Emulator (QEMU)
- Libvirt

Key facts:

- Real-time KVM is, for instance used in virtual Radio Access Network (vRAN) when low average predictable latency is more important than fast performance.

1.2.3. OpenStack Layer

The OpenStack layer is mainly composed of Red Hat OpenStack Platform. Only some of the OpenStack services are relevant in an NFV use case. Red Hat, as a top OpenStack contributor, collaborates with other leaders in the industry of NFV features, like Intel, Cisco, Huawei, NEC, etc. in the upstream OpenStack community. The main services requiring NFV special features are:

- Compute (nova)
- OpenStack Networking (neutron)
- Block Storage (cinder)
- Telemetry (ceilometer)

Key facts:

- Compute NFV features such as vCPU Pinning, Large Pages, and NUMA awareness.
- OpenStack Networking NFV features such as IPv6, SR-IOV, and integration with DPDK-accelerated Open vSwitch.
- Block Storage service is main storage service relevant to NFV. It provides carrier grade storage requirement with Ceph as a back-end.

- Telemetry, OpenStack’s metering service, collects events in a dedicated database. These events are then sent to management tools for further consumption. FCAPS is an acronym for fault, configuration, accounting, performance, security, the management categories into which the ISO model defines network management tasks.

1.2.4. Management Layer

The Management layer is mainly composed of:

- Red Hat CloudForms
- Red Hat OpenStack Platform director
- FCAPS operational tools

Key facts:

- Red Hat CloudForms is the cloud management tool that presents data from multiple sources like the Virtual Infrastructure Manager (VIM) and the Network Functions Virtualization Infrastructure (NFVI) in a unified display. It allows you to manage hybrid VIM scenario (VNF running on legacy virtualization platform and VNF running on top Openstack platform) to consolidate FCAPS.
- Red Hat OpenStack Platform director allows for provisioning of the NFV infrastructure (NFVi), to support a wide range of telecom applications such as VNFs. It allows for update and upgrade.
- Operational tools provide fault, configuration, accounting, performance, security (FCAPS) management. These tools allow you to monitor the health of the NFVi. These are open source tools that are used by cloud community to manage cloud platform at scale.

2. TECHNOLOGY DETAILS

The following features are used in the Red Hat NFV Solution configuration.

2.1. Compute

Compute (nova) manages instances in an OpenStack environment by creating, scheduling, and deactivating virtual machines as necessary. Compute is relevant to NFV use cases. For example, the scheduler is crucial for performance and resiliency. This can be achieved by placing instances correctly. It must also launch new instances quickly, both initially and especially in reaction to fault detection. Some of the key Compute features are described below:

2.1.1. Non Uniform Memory Access (NUMA) Topology awareness

In 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. Each CPU socket can have multiple CPU cores which are treated as individual CPUs for virtualization purposes.

OpenStack Compute (nova) 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). For more information, see NUMA.

2.1.2. vCPU Pinning

CPU pinning is the ability to run a specific virtual machine’s virtual CPU on a specific physical CPU, in a specific host.

An OpenStack Compute environment can support the pinning of virtual machine instances to dedicated physical CPU cores. To facilitate this, the steps are as follows:

- Reserve dedicated cores on the compute hosts for host processes
- Reserve dedicated cores on the compute hosts for the virtual machine instances themselves
- Enable the required scheduler filters
- Create a host aggregate to add all hosts configured for CPU pinning
- Create a performance focused flavor to target this host aggregate
- Create a host aggregate for all hosts that are not configured for CPU pinning and map all other flavours to it.
- Launch an instance with CPU pinning

For more information, see Configure CPU Pinning with NUMA.

2.1.3. Large-pages (or Huge Pages) support

Physical memory is segmented into a series of contiguous regions called pages. To improve efficiency, instead of accessing individual bytes of memory, the system retrieves memory by accessing entire pages. Each page contains number of bytes, referred to as the page size. To access a page, the system first translates the virtual addresses into physical addresses to determine which page contains the requested memory.

At a basic level, processes that use large amounts of memory and/or are otherwise memory intensive, may benefit from larger page sizes, often referred to as large pages or huge pages. Red Hat Enterprise Linux 7.1 systems support 2 MB and 1 GB huge pages, which can be allocated at boot or at runtime. For more information, see Huge Pages and Transparent Huge Pages.

2.1.4. Support (PCIPassThrough)

PCI passthrough allows guests to have exclusive access to PCI devices for a range of tasks. Standard passthrough allows virtual machines exclusive access to PCI devices and allows the PCI devices to appear and behave as if they were physically attached to the guest OS. In the case of networking, it is possible to utilize PCI passthrough to dedicate an entire network device (for example, a physical port on a network adapter) to a guest OS running within a virtual machine.

Single Root I/O Virtualization (SR-IOV) is a specification that allows a PCI device to separate access to its resources among various PCI hardware functions.

For more information, see NUMA Node Locality for PCI Devices.

2.2. Networking

The performance requirement is largely about high-performance packet processing - how to get a packet off the network, into a virtual machine, processed quickly and back out again on the network. One of the available techniques is to give virtual machines direct physical access to the network via SR-IOV. OpenStack Networking has supported IPv6 tenant subnets in certain configurations. DPDK-accelerated Open vSwitch for fast packet processing with accelerated virtual switches are also supported in OpenStack Networking as a technology preview in the Red Hat OpenStack Platform 8 release.

2.2.1. Single Root I/O Virtualization (SR-IOV)

SR-IOV enables a single root function (for example, a single Ethernet port), to appear as multiple, separate, physical devices. A physical device with SR-IOV capabilities can be configured to appear in the PCI configuration space as multiple functions. Each device has its own configuration space complete with Base Address Registers (BARs). SR-IOV uses two PCI functions:

- Physical Functions (PFs) are full PCIe devices that include the SR-IOV capabilities. Physical Functions are discovered, managed, and configured as normal PCI devices. Physical Functions configure and manage the SR-IOV functionality by assigning Virtual Functions.
- Virtual Functions (VFs) are simple PCIe functions that only process I/O. Each Virtual Function is derived from a Physical Function. The number of Virtual Functions a device may have is limited by the device hardware. A single Ethernet port, the Physical Device, may map to many Virtual Functions that can be shared to virtual machines.

The hypervisor can map one or more Virtual Functions to a virtual machine. The Virtual Function’s configuration space is then mapped to the configuration space presented to the guest.

OpenStack Networking can put aside the previous requirement for virtual bridges, and extend the physical NIC’s capabilities directly through to the instance. For more information, see SR-IOV Support for Virtual Networking.

2.2.2. Port-security extension

Previously, as a security measure, OpenStack Networking security group implementation always applied the default rules automatically to block IP address spoofing attacks, preventing virtual machines from sending or receiving traffic with MAC or IP address which does not belong to its ports.

In NFV use cases, where network functions are running within virtual machines, the option to turn off the default anti-spoofing rules is necessary. This extension allows administrators to enable or disable the security-group feature per port using the ‘port-security-enabled’ attribute, so that the tenant admin can decide if and where a firewall is needed in the topology.

2.2.3. IPv6 support

OpenStack Networking supports IPv6 tenant subnets in dual-stack configuration, so that projects can dynamically assign IPv6 addresses to virtual machines using Stateless Address Autoconfiguration (SLAAC) or DHCPv6. OpenStack Networking is also able to integrate with SLAAC on your physical routers, so that virtual machines can receive IPv6 addresses from your existing infrastructure. For more information, see Tenant Networking with IPv6.

2.2.4. DPDK-accelerated vSwitch

DPDK consists of a set of libraries and user-space drivers for fast packet processing, enabling applications to perform their own packet processing directly to/from the NIC, delivering up to wire speed performance for certain use cases. DPDK-accelerated Open vSwitch (OVS+DPDK) significantly improves the performance of Open vSwitch while maintaining its core functionality. It enables the packet switching from the host’s physical NIC to the application in the guest instance (and between guest instances) to be handled almost entirely in user-space.

OpenStack Networking (neutron) OVS plugin is updated to support OVS+DPDK back end configuration. For more information, see Configure DPDK for OpenStack Networking.

DPDK-accelerated vSwitch feature is introduced as a technical preview in the Red Hat OpenStack Platform 8 release.

2.2.5. Quality of Service (QoS)

QoS refers to the resource control systems that guarantees certain network requirements such as, bandwidth, latency, reliability in order to satisfy a Service Level Agreement (SLA) between an application provider and end users. Network devices such as switches and routers can mark traffic so that it is handled with a higher priority to fulfill the QoS conditions agreed under the SLA. In other cases, certain network traffic such as Voice over IP (VoIP) and video streaming needs to be transmitted with minimal bandwidth constraints. On a system without network QoS management, all traffic will be transmitted in a “best-effort” manner making it impossible to guarantee service delivery to customers.

Starting with Red Hat OpenStack Platform 8, egress bandwidth limit rules are supported, so that virtual machines can be rate-limited. For more information, see Configure Quality-of-Service (QoS).

2.2.6. OpenDaylight

OpenDaylight delivers a common open source framework and platform for Software-Defined Networking (SDN) across the industry for customers, partners and developers. SDN enables users to program network layers, separating the data plane from the control plane while NFV allows for agile placement of networking services when and where they are needed. By enabling this level of programmability, SDN and NFV can enable users to optimize their network resources, increase network agility, service innovation, accelerate service time-to-market, extract business intelligence and ultimately enable dynamic, service-driven virtual networks.

OpenDaylight is supported as Technical Preview starting in the Red Hat OpenStack Platform 8 release. Red Hat’s OpenDaylight distribution is OpenStack-centric, and only ships the required interfaces to OpenStack Networking and to Open vSwitch (via OVSDB NetVirt).

2.3. Storage

OpenStack storage plays an important part of data protection and disaster recovery to the cloud. Red Hat OpenStack Platform allows customers and users to securely backup their critical data onto the cloud storage as well as address their compliance requirements for off-site backups and benefit from the lower cost offered by cloud storage while reducing the cost in maintaining a secondary cloud for disaster recovery.

2.3.1. Volume Migration

OpenStack has the ability to migrate volumes between backends which support its volume-type. Migrating a volume transparently moves its data from the current back-end for the volume to a new one. Volume migratio allows a better way to manage the volume migration status, adding the migration progress indication, enriching the notification system by reporting to Ceilometer, guaranteeing the migration quality with tempest tests and CI support, etc. For more information, see Migrate a Volume.

2.3.2. Snapshot Import and Export

Provides a means to import/export snapshots. The ‘export’ allows you to migrate a snapshot from one Block Storage volume to another, while the ‘import’ allows you to import non-OpenStack snapshots on a back-end device into OpenStack or Block Storage volumes. The advantages are as follows:

- You can import snapshots as volume templates to create volumes from snapshots.
- You can also import snapshots to provide an effective means to manage the import volumes.

2.4. High Availability

High Availability refers to a system or component that is continuously operational for a long length of time which include enough excess capacity in the design to accommodate a performance decline or a subsystem failure. Availability can be measured relative to "100% operational" or "never failing." A widely-held but difficult-to-achieve standard of availability for a system or product is known as "five 9s" (99.999 percent) availability. A highly available system seeks to minimize the system downtime, and data loss.

High availability is implemented by adding redundant hardware running redundant instances of each service. If one piece of hardware running an instance of service fails, the system can then failover to use another instance of the service.

2.4.1. High Availability (keepalived)

Keepalived runs on an active Linux Virtual Servers (LVS) router as well as one or more optional backup LVS routers. The active LVS router serves two roles:

- To balance the load across the real servers.
- To check the integrity of the services on each real server.

The active (master) router informs the backup routers of its active status using the Virtual Router Redundancy Protocol (VRRP), which requires the master router to send out advertisements at regular intervals. If the active router stops sending advertisements, a new master is elected.

The **`keepalived`** daemon is used to monitor services or systems and to automatically failover to a standby if problems occur. The **`keepalived`** daemon implements a set of checkers to dynamically and adaptively maintain and manage load-balanced server pool according their health. It implements a set of hooks to the Virtual Router Redundancy Protocol (VRRP) finite state machine providing low-level and high-speed protocol interactions. The **`keepalived`** frameworks can be used independently or all together to provide resilient infrastructures.

2.4.2. Instance High Availability (pacemaker remote)

Pacemaker is the state-of-the-art high availability and load balancing stack for the Linux platform. Instance high availability is achieved by monitoring the virtual machines for failure and trigger an automatic response for fencing and recovery. Fencing is an operation that completely isolates a failed host. Recovery system orchestrates the rescue of virtual machines from the failed host.

Pacemaker is useful to make OpenStack infrastructure highly available. It does not inherently understand the applications it manages. Instead, it relies on resource agents (RAs). Resource agents are scripts that encapsulate the knowledge of how to start, stop, and check the health of each application managed by the cluster.

There are three key capabilities to any solution endeavoring to provide workload high availability in a cloud or virtualization environment:

- A monitoring capability to detect when a given compute node has failed and trigger handling of the failure.
- A fencing capability to remove the relevant compute node from the environment.
- A recovery capability to orchestrate the rescuing of instances from the failed compute node.

2.5. Monitoring

Monitoring a system ensures optimum performance. The data collected can be used for customer billing, system monitoring, or alerts. Knowing that failures are bound to occur, OpenStack needs to focus on resiliency monitoring, fault detection and response.

With OpenStack, monitoring the metrics and events listed below will provide good visibility into the health and performance of your deployment:

- hypervisor_load
- running_vms
- free_disk_gb
- free_ram_mb
- queue memory
- queue consumers
- consumer_utilisation

You can monitor additional metrics that are particularly relevant to the infrastructure and use cases, and what you monitor will depend on both the operational tools (FCAPS) you have and the OpenStack components you are using.

FCAPS tools allow you to manage fault, configuration, accounting, performance and security issues. The goals of the FCAPS management tools are as follows:

- A fault is an event that has negative significance. Fault management recognises, isolates, corrects and logs faults that occur in a network. It uses trend analysis to predict errors so the network in highly available.
- Configuration management monitors system configuration information, and any changes that take place. This is especially important, since many network issues arise as a direct result of changes made to configuration files, updated software versions, or changes to system hardware.
- Accounting management tracks network utilization information, such that individual users, departments, or business units can be appropriately billed or charged for accounting purposes. For non-billed networks, "administration" replaces "accounting". The goals of administration are to administer the set of authorized users by establishing users, passwords, and permissions, and to administer the operations of the equipment such as by performing software backup and synchronization. Accounting is often referred to as billing management. Using the statistics, the users can be billed and usage quotas can be enforced. These can be disk usage, link utilization, CPU time, etc.
- Performance management focuses on ensuring that network performance remains at acceptable levels. It enables the manager to prepare the network for the future, as well as to determine the efficiency of the current network, for example, in relation to the investments done to set it up. The network performance addresses the throughput, network response times, packet loss rates, link utilization, percentage utilization, error rates and so forth.
- Security management controls access to assets in the network. Data security can be achieved mainly with authentication and encryption. Security management is not only concerned with ensuring that a network environment is secure, but also that gathered security-related information is analyzed regularly. Security management functions include managing network authentication, authorization, and auditing, such that both internal and external users only have access to appropriate network resources. Other common tasks include the configuration and management of network firewalls, intrusion detection systems, and security policies (such as access lists).

2.6. Real-time KVM

Existing hypervisors are not designed or targeted to meet the requirements for NFVi, and further collaborative efforts are needed to enable NFV features. KVM allows for high-performance and low latency. Different uses cases have different allowances for latency, but many require less than 20 microseconds latency. Some even require 5-10 microseconds and that performance must be guaranteed. To achieve these numbers, the system needs to be carefully set up to avoid all kinds of high-latency system operations, for example, no CPU frequency changes, no loading or unloading of kernel modules, and no swapping. The applications also have to be tuned to avoid slow devices, for example, disks or sound devices except in non-realtime helper programs. So deploying real-time KVM requires deep knowledge of the system, for example, to ensure the timestamp counter is stable and the system will never fall back to another clock source. Using Real-time KVM allows higher latency with lower peaks and the ability to advertise an expected latency band.

Processing performance and latencies depend on a number of factors, including the CPUs (frequency, power management features, etc.), micro-architectural resources, the cache hierarchy and sizes, memory (and hierarchy, such as NUMA) and speed, inter-connects, I/O and I/O NUMA, devices, etc.

The following are initial enhancements to Real-time KVM for NFV:

- Minimal Interrupt latency variation for data plane VNFs
    - Minimal Timing Variation for Timing correctness of real-time VNFs can be achieved by timing correctness for scheduling operations, such as Radio scheduling.
    - Minimal packet latency variation for data-plane VNFs which can be achieved by packet delay variation, which applies to packet processing.

For a virtual machine, interrupt latency (time between arrival of the hardware interrupt and invocation of the interrupt handler in the virtual machine, for example, can be either the timing variation, or packet latency, or both, depending on the type of the device. Interrupt latency with a virtual timer can cause timing correctness issues with real-time VNFs even if they only use polling for packet processing. The VNFs need to be implemented properly to minimize interrupt latency variation within the virtual machines.

Real-time KVM is supported as Technical Preview starting in the Red Hat OpenStack Platform 8 release

3. TERMINOLOGY

This section discusses some of the terms used in this article.

3.1. Open Source

Open Source is part of the Red Hat DNA. The two key requirements from a CSP when starting the NFV journey are rapid innovation and prevention of vendor lock-in. Open source allows you to achieve this. The notion of open-source is linked with the commitment to submit any new NFV features to the upstream community first. The benefit is, you have no hidden maintenance costs to support proprietary features.

Open Platform for NFV (OPNFV) provides an open source platform for deploying NFV solutions that leverages investments from a community of developers and solution providers. OPNFV is a carrier-grade, integrated, open source platform to accelerate the introduction of new NFV products and services. Red Hat is an active member of OPNFV project.

3.2. Carrier Grade

Over time, telecom service providers have engineered an extensive range of features into their networks where they can guarantee ‘six-nines’ reliability. This means that the network is guaranteed to be up 99.9999% of the time, implying a downtime of no more than 32 seconds per year. As service providers refine their plans to progressively introduce NFV into their networks, they are reviewing the implications for six-nines reliability on network infrastructure that incorporates virtualized functions. The list of requirements for achieving carrier-grade reliability are availability, security, performance and management.

Red Hat’s strategy for the Red Hat Solution is to work with the customers and partners to develop features to make the Red Hat solution carrier grade in an open source way with our upstream first policy. This means that Red Hat works to get features integrated into open source projects before we integrate them into our product offerings. In the context of VNF that are deployed in an Openstack environment, the term carrier grade needs to be redefined and refined with the maturity of the technology. We have already developed multiple features in networking performance, high availability, manageability, ease of deployment and use. But the development is not complete, together with Intel, Red Hat’s objective is to solve the remaining gaps and achieve operation and production readiness so that CSP can deploy VNF with similar confidence that they used to with legacy hardware.

3.3. Integrated Engineering

To build a complete opensource NFV solution, you have to integrate multiple components coming from very different communities (Openstack, KVM, Linux, etc.). Red Hat is a key contributor in these communities and proposes a commercial solution that integrates all components. This is key as it allows the following:

1. To develop new NFV features. For instance, developing a new NFV feature often requires modification and synchronization on Linux, KVM, Libvirt, OpenStack Compute and Openstack Networking services.
2. To support these NFV features. The team responsible for the development of these features are often responsible for supporting these features. This is the devops concept. Guaranteeing carrier grade SLA requires both community support and experts who provide commercial support.

3.4. European Telecommunication Standards Institute (ETSI)

The ETSI is an independent standardisation group that develops standards for information and communications technologies (ICT) in Europe. ETSI Industry Specification Group for Network Functions Virtualization (ETSI ISG NFV) is in charge of developing requirements and architecture for virtualization for various functions within telecom networks.

Network functions virtualization (NFV) focuses on addressing problems involved in using proprietary hardware devices. By evolving standard IT virtualization technology, NFV implements network functions into software so that it can run on a range of industry standard server hardware and can easily be moved to various locations within the network as needed. With NFV, the necessity to install new equipment is eliminated. The ETSI ISG NFV helps by setting requirements and architecture specifications for hardware anc software infrastructure needed to make sure virtualized functions are maintained. ETSI ISG NFV also manages guidelines for developing network functions.

3.5. Management and Orchestration (MANO)

The NFV MANO architecture is responsible for managing cloud infrastructure, deploying and connecting hosted elements or VNFs and services composed of VNFs.

3.6. Virtual Evolved Packet Core (vEPC)

vEPC is a cloud architecture that provides a platform for the core components of Long-Term Evolution (LTE) network, allowing dynamic provisioning of gateways and mobile endpoints to sustain the increased volumes of data traffic from smartphones and other devices.

3.7. Virtual IP Multimedia Subsystem (vIMS)

vIMS facilitates the use of IP (Internet Protocol) for packet communications in all known forms over wireless or landline.