Tuning VMware vCloud NFV for Data Plane Intensive Workloads

Technical White Paper

vmware®

Table of Contents

VMware, Inc. 3401 Hillview Avenue Palo Alto CA 94304 USA Tel 877-486-9273 Fax 650-427-5001 www.vmware.com

Introduction

Since the first release of VMware vCloud® NFV™, many new features and capabilities have been added to the solution. As more Communication Service Providers (CSPs) adopt vCloud NFV, and as an increasing number of network functions are virtualized, the VMware NFV solutions team gains experience in the operation of NFV environments hosting a variety of Virtual Network Functions (VNFs). Our team has performed extensive platform tests to provide our partners and customers guidance on tuning the Network Functions Virtualization Infrastructure (NFVI) for optimal VNF performance. This real-world and testing-based expertise are summarized in this best practice document.

This white paper addresses one type of telecommunications workload: the data plane. Examples of such workloads include the Packet Gateway (PGW) in the mobile packet core, Session Border Controller (SBC) in a Voice over Long-Term Evolution (VoLTE) deployment, Broadband Network Gateway (BNG) in a residential access environment, and the service provider MPLS Provider Edge (PE) router. In their hardware incarnation, these functions often challenged the vendors developing them to increase Network Processing Unit (NPU) power and density, invent new interfaces, and develop tight integration between hardware and software. Such workloads were bound inextricably to hardware implementations until the inception of NFV.

Traditional data plane intensive workloads are bound by two aspects – packet processing and packet forwarding. Packet processing refers to the functionality and logic a network function delivers and is usually handled by purpose-built hardware. Packet forwarding refers to the process of packets arriving to a network function (ingress) and packets leaving the network function (egress). When migrating data plane intensive network functions from purpose-built hardware to the virtual domain, the server's general purpose CPU is used for packet processing, and for ingress and egress packet forwarding.

Tuning and optimizing the VNF's packet processing and packet forwarding are the subject of this white paper. Best practice concrete recommendations are provided to help tune the vCloud NFV to optimally serve data plane intensive workloads. Following the recommendations described here, the audience will be able to reach optimized data plane performance, supporting real-world requirements for use cases such as Mobile Packet Core, Gi-LAN services, MPLS PE routers, and more.

Audience

This document assumes the reader is versed in the NFVI concepts described by the European Telecommunications Standards Institute (ETSI) Network Functions Virtualization (NFV) Industry Specification Group (ISG). It is written for two specific audiences: the vendors creating VNFs and the CSPs looking to deploy data plane intensive VNFs. This paper provides vendors with a clearer understanding of how to best design VNFs to suit the needs of CSPs, and the configuration discussed here enables CSPs to design the NFVI to best suit data plane intensive workloads.

This paper covers VMware vCloud NFV versions 1.5 through 2.0.

vmware®

VMware, Inc. 3401 Hillview Avenue Palo Alto CA 94304 USA Tel 877-486-9273 Fax 650-427-5001 www.vmware.com

NFV Workload Classification

The ETSI Network Functions Virtualisation (NFV); NFV Performance & Portability Best Practices (ETSI GS NFV-PER 001) document separates NFV workloads into different classes. At a high level, the characteristics distinguishing the workload classes are:

- **Data Plane Workloads**. Data plane workloads include all operations relating to end-to-end communication packet handling between end points. These tasks are expected to be very I/O intensive in operations, involving numerous memory read-write operations.
- **Control Plane Workloads**. Control plane workloads reflect communication between network functions (NFs) required for these functions to perform their tasks. Examples of control plane workloads include routing, signaling, session management, and session authentication. When compared to data plane workloads, control plane workloads are expected to be much less intensive in their packet per second handling requirements, while the complexity of such transactions can be higher. These workloads often must adhere to certain transaction delay requirements, which dictate that maintaining low latency is important.
- **Signal Processing Workloads**. These workloads include all tasks related to digital processing, such as decoding and encoding in a cellular base station. These tasks are expected to be very intensive in CPU processing capacity, and highly delay sensitive.
- **Storage Workloads**. The storage workloads include all tasks related to disk storage.

Control plane workloads have been adequately supported using standard out of the box configurations as described in the VMware vCloud NFV Reference Architecture. Data plane and signal processing workload performance can benefit from further tuning. Storage workloads are beyond the scope of this paper.

VMware vCloud NFV Architecture

VMware vCloud NFV is an ETSI compliant, fully integrated, modular, and multitenancy NFV platform. vCloud NFV enables wireline and wireless service providers to deploy an elastic business model for cross-cloud services and service enablement, while simplifying networks and reducing total cost of ownership.

In the design of vCloud NFV, the management components required to operate an environment are separated from the NFVI where VNFs are deployed. This ensures that the common resources, available in the servers, are dedicated only to the network functions they support. It also empowers the CSP to deploy powerful servers and high-performing network interface cards (NICs) where data plane intensive VNFs reside, while reducing the hardware footprint for less demanding VNFs.

Virtual Network Functions, Components, and Virtual Machines

This white paper details the approach and configuration required to optimize the internal components of a VNF, and the NFV infrastructure on which the VNF is deployed, for data plane performance. VNF internal components, defined by ETSI as Virtualized Network Function Components (VNFCs), provide certain necessary characteristics for the functioning of the VNF. Each VNFC maps to a single virtualization container, or virtual machine (VM). According to the ETSI Network Functions Virtualisation (NFV); Terminology for Main Concepts in NFV (ETSI GS NFV 003), a VM is capable of hosting a VNFC. In this document, the terms VM and VNFC are used interchangeably.

Understanding the Data Plane Path

The speed of a network is determined by the its slowest path. The data path in the virtual domain follows the same principle, which is why understanding the data plane path in the virtual domain is essential to performance tuning. The data path in the virtualization domain relies on the compute domain, which consists of physical components such as network interface cards, processors (CPUs), memory, and buses. Both domains must be considered and tuned holistically.

As Figure 1 depicts, from a VNFC (VM) perspective, the underlying VMware ESXi™ hypervisor is responsible for two network functions: delivering traffic to the VNFC and sending traffic from the VNFC out to the network. The hypervisor is also responsible for providing the VNF with the resources it needs to process network traffic. The network path consists of a physical NIC (pNIC), a process which transports traffic from the pNIC to the VM, and a process that sends traffic from the VM to the network.

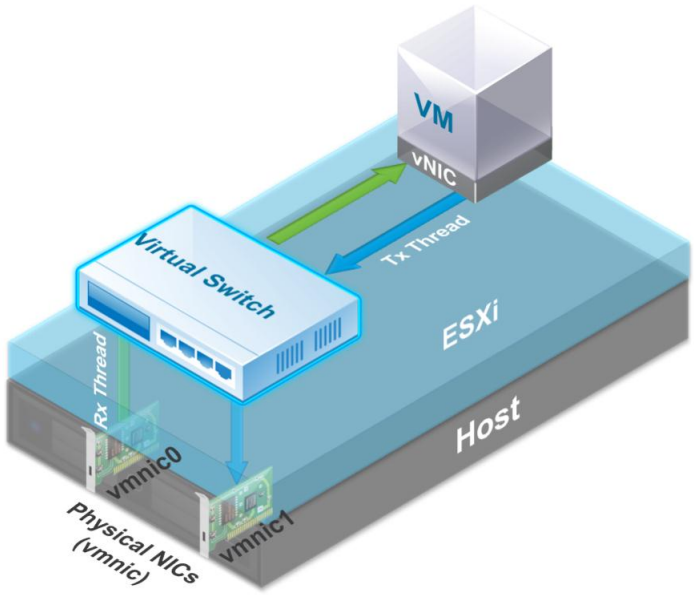

Figure 1: Data Path to a Virtual Machine

The process of receiving traffic from the pNIC to a virtual machine is called the Rx thread. To understand the use of an Rx thread, it is important to first follow the way in which network traffic is received by the network interface cards (NICs) and then transferred to the virtual machine.

Modern NICs support multiple hardware queues. ESXi uses a logical queue concept known as NetQueue to distribute and load balance hardware queues to multiple CPUs. By default, ESXi NetQueue enables MAC address filtering on physical NICs, so incoming packets with the same destination MAC address are mapped to the same hardware queue, and are processed by an individual Rx thread. For example, a virtual machine with two vNICs (or two virtual machines on the same ESXi host, each with its own vNIC) receiving packets, will have two separate Rx threads, each mapped to a different hardware queue. For VLAN based unicast traffic, the default NetQueue behavior is the recommended configuration for improving the receiving side of packet performance.

Transmitting packets from the virtual machine is achieved by utilizing a transmit thread, or Tx thread. A virtual machine on ESXi can have between 1 and 10 virtual NICs (vNICs) connected to one or more virtual switch (vSwitch) port groups. The transmit thread executes all parts of the virtual networking stack including the vNIC emulation (such as VMXNET3), the vSwitch, the uplink layer, and packet scheduling to the physical NIC (pNIC) driver. If the vSwitch determines that a packet is destined for another VM on the same host, the transmit thread also handles the tasks for receiving the packet on the destination VM's vNIC port and emulating that destination vNIC's receive operation. By default, there is only one transmit thread for a given virtual machine, regardless of the number of vNICs. However, this behavior can be configured such that multiple transmitting threads are created for a virtual machine, as described in detail later in this document.

Virtual Network Function Design

Many of the data plane intensive VNFs available on the market today are rooted in years of development on specialized hardware. As these network functions migrate from physical, purpose-built hardware, to the virtual domain, there are a few steps that must be taken to arm them with the capabilities required to achieve optimal performance.

Data plane intensive VNFs often use similar architecture. These VNFs include a VNFC responsible for load balancing traffic incoming and outgoing to and from the VNF; packet processing components responsible for terminating sessions; encapsulating and decapsulating traffic and making forwarding decisions; and Operations, Administration, and Management (OAM) functions. The latter is typically light on packet processing, whereas the first two types of components are packet-processing intensive.

The focus of VNF design is to:

- Identify the VNF Component configuration needed to achieve certain packet processing targets
- Identify the physical resources required to achieve the same packet processing targets
- Address any limitations on the VNF resulting from achieving the throughput targets
- Validate that the VNF packet processing capabilities can be scaled up in a deterministic way

Once all VNF Components have been tuned for performance and the VNF itself is created, the overall performance of the VNF must be measured in relationship to the type of traffic it is expected to handle. Sizing recommendations must be captured and provided to communication service provider customers, so they can ensure that their NFV infrastructure resources are available for the intended performance of the VNF.

This completes the VNF design process as depicted in Figure 2.

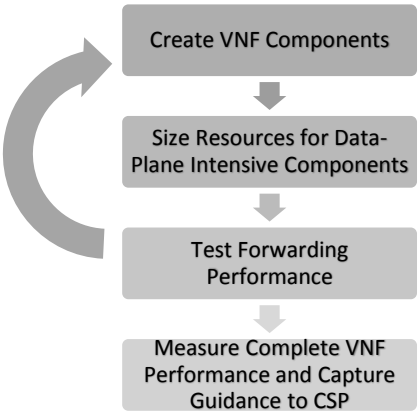

Figure 2: The Virtual Network Function Design Process

Virtual Network Function Component Creation

There are many aspects to developing VNFs and their components. These aspects, such as standards compliancy, code development, and roadmap design, are the responsibility of the VNF vendor. In this section of the white paper, important aspects of VNF development are explained as they pertain specifically to running data plane intensive VNFs on the vCloud NFV platform.

Data Plane Development Kit

The open source project Data Plane Development Kit (DPDK) is a set of libraries and drivers for fast packet processing. Since DPDK most often runs in Linux, it can be used by VNF vendors to improve data plane performance. DPDK performance improvements are available with every release of the libraries, so taking the DPDK release cycles into account in your VNF development plan and incorporating the latest DPDK code into the VNF is advisable.

Using the VMware VMXNET3

VMware created the paravirtualized network interface card (NIC) VMXNET3 for improved performance over other virtual network interfaces. This vNIC provides several advanced features including multi-queue support, Receive Side Scaling (RSS), Large Receive Offload (LRO), IPv4 and IPv6 offloads, and MSI and MSI-X interrupt delivery. By default, VMXNET3 also supports an interrupt coalescing algorithm. Virtual interrupt coalescing helps drive high throughput to VMs with multiple vCPUs with parallelized workloads, for example with multiple threads, while at the same time striving to minimize the latency of virtual interrupt delivery.

A data plane intensive VNFC must always use VMXNET3 as its vNIC rather than using other interfaces, such as an E1000 adapter. If a VNFC uses DPDK, a VMXNET3 poll mode driver must be used. VMXNET3 has been included in several modern Linux distributions such as Red Hat Enterprise Linux and SUSE Linux since its release 2.6.32 in October of 2009. Although the VMXNET3 driver is included in the Linux distributions, VMware Tools must also be installed on the VM. See the VMware Knowledge Base article VMware support for Linux inbox VMware drivers (2073804) for more information. For further information about DPDK support in the VMXNET3 driver, please refer to the article Poll Mode Driver for Paravirtual VMXNET3 NIC.

Virtual Machine Hardware Version

It is important to ensure a VNFC uses the most up-to-date virtual machine hardware version. The virtual machine hardware version reflects the supported virtual hardware features of the VM. These features correspond to the physical hardware available on the ESXi host on which the VM is created. Virtual hardware features include the BIOS and Extensible Firmware Interface (EFI), available virtual PCI slots, maximum number of CPUs, maximum memory configuration, and other characteristics typical to hardware.

Uniform VM hardware versions must be used for all VMs composing a VNF. This is especially important since different VM hardware versions support different components and different amounts of resources. For example, VM hardware version 13 supports a maximum of 6128 GB of RAM for a VM, whereas VM hardware version 11 supports 4080 GB of RAM. VM hardware versions also enable processor features, which is why the best practice recommendation is to use the latest virtual hardware version to expose new instruction sets to the VM. Mismatches between VNFCs configured with different VM hardware versions impact performance and must be avoided. Hardware version differences are listed in the Knowledge Base article Hardware features available with virtual machine compatibility settings (2051652).

The latest VM virtual hardware versions and their matching hypervisor versions are listed in the Knowledge Base article Virtual machine hardware versions (1003746). For vCloud NFV versions 1.5 and 1.5.1, which

use vSphere 6.0 U2, virtual machine hardware version 11 is recommended. For vCloud NFV 2.0, which uses vSphere 6.5, virtual machine hardware version 13 is recommended. Upgrading hardware versions is a simple process described in Knowledge Base article Upgrading a virtual machine to the latest hardware version (multiple versions) (1010675).

Recommendations:

1. Compile your VNFC with the latest DPDK version.
2. Install and use VMXNET3 adapters to your VNFCs.
3. Configure your VNFCs to use the latest hardware version and ensure that all VNFCs in a VNF use the same hardware version.

Virtual Network Function Component Configuration for Optimal Throughput Performance

Once the VNFCs are created, the following recommendations must be evaluated to ensure that adequate resources are available to achieve optimal performance.

Identifying the number of virtual CPUs a VNFC requires to perform its role is a crucial performance investigation that must be undertaken together with an evaluation of the amount of CPU cores required to deliver packets to the VNFC and send packets back out. The next sections of this document discuss techniques and proper configuration to reserve CPU cycles and CPU cores for a specific process or component. After these recommendations are followed, the number of CPU cores required for optimal data plane performance can be evaluated.

NUMA Node Alignment

A key guideline to achieving optimal performance in a data plane intensive VNFC is to remain within the NUMA boundaries. A NUMA node is defined as the set of CPUs and the range of memory that are reported to be within a proximity domain by the firmware. Normally, the CPUs and memory range are connected on the same bus, as a result providing low-latency local memory access. On a server with multiple NUMA nodes, an interconnect such as Intel QPI exists, to connect among pairs of NUMA nodes.

One can expect improved networking performance when a complete vertical alignment of components to NUMA nodes exists. This includes not only the physical NIC, but all vCPUs, memory, and threads associated with a VNFC must be scheduled on the same NUMA node ID.

Processor affinity for vCPUs to be scheduled on specific NUMA nodes, and memory affinity for all VM memory to be allocated from those NUMA nodes, can be set using the VMware vSphere® Web Client.

Follow these steps:

1. In the vSphere Web Client go to the **VM Settings** tab.
2. Choose the **Options** tab and click **Advanced General** > **Configuration Parameters**.
3. Add entries for **`numa.nodeAffinity=0, 1, ...,`** where `0` and `1` are the processor socket numbers.

VMware, Inc. 3401 Hillview Avenue Palo Alto CA 94304 USA Tel 877-486-9273 Fax 650-427-5001 www.vmware.com

Recommendations:

1. Ensure that your VNFC and the resources used to support it fit within a NUMA node.
2. When possible, strive for complete vertical alignment between network interfaces, memory, physical CPUs, and virtual machines to a single NUMA node.

Assigning Exclusive CPU Cores

As described in the *Understanding the Data Plane Path* section of this document, there are several functions associated with a data plane intensive VNFC. The component must receive packets from the physical NIC, it must process the packets, performing the task for which it was designed, then it must send the packets out toward the egressing NIC. These three processes must be understood, tuned, and sized, based on the VNF characteristics.

The logical function to investigate first is the VNFC itself. If a VNFC is unable to process more packets than the underlying layer is providing, no further optimization is required. When a VNFC is observed to be losing packets, and its vCPU's load is measured at 100%, avoiding context switching can potentially help increasing the performance of the VNFC.

One of the major benefits of a virtualized environment is shared resources. With VMware vSphere® Distributed Resource Scheduler™, VNFCs receive resources as needed. Sharing CPU resources between different processes requires switching between one process and another. During that time, when a data plane intensive process is in place, the CPU may lack the necessary cycles to serve network traffic. This may cause packet loss to the data plane intensive workload.

To address this, a VNFC's vCPU can be pinned to a CPU core and never be moved to a different core. In addition, the same vCPU can be configured to receive exclusive CPU affinity, which will block any other vCPU from accessing the core. Both vCPU pinning and exclusive affinity are needed to eliminate context switching. If, however, the CPU is not constantly used by the VNFC, these CPU resources are wasted. This engineering trade-off must be considered during the testing and preparation phase of the VNF.

Dedicating CPUs to a VNF Component

To ensure pinning and exclusive affinity of all vCPUs of a VNFC, set the VM configuration parameter **`sched.cpu.latencySensitivity`** to **High**. This configuration will also disable interrupt coalescing automatically, which is typically not an issue if the VNFC is compiled with DPDK and uses a VMXNET3 DPDK poll mode driver. If the VNFC is not compiled with DPDK, the best practice recommendation is to manually re-enable interrupt coalescing.

To set the VNFC configuration for CPU affinity, follow these steps:

- In the vSphere Web Client go to **VM Settings**.
- Choose the **Options** tab and select **Advanced General** > **Configuration Parameters**.
- Enter **High** as the value of **`sched.cpu.latencySensitivity`**.

Before using this configuration parameter, the VM CPU reservation must be set to its maximal value and the CPU limit must be set to **Unlimited.**

VMware, Inc. 3401 Hillview Avenue Palo Alto CA 94304 USA Tel 877-486-9273 Fax 650-427-5001 www.vmware.com

To manually re-enable interrupt coalescing, follow these steps:

1. Edit the **VMX Options**.
2. Set the **`ethernetX.coalescingScheme`** to **rbc** where X is replaced with the NIC number.
3. Replace the **`X`** with the number of the vNIC used.

Improving Parallel Processing for Data Plane Intensive Components

By default, ESXi uses one Tx thread regardless of the number of vNICs associated with a VNFC. When the component sends more traffic to the vNICs than a single Tx thread can handle, one can achieve improved parallel processing by assigning a Tx thread to each vNIC. When adding the command **`ethernetX.ctxPerDev = "1"`** where **`X`** is replaced with the vNIC number, each vNIC configured with **`ctxPerDev`** will receive a Tx thread. The naming scheme for vNICs uses the convention of **`ethX`** where **`X`** is a numerical value. That value is the vNIC number. It is important to keep in mind that the Tx thread is only made available when traffic is egressing the VNFC.

VNFCs can be configured with several vNICs, however typically not all vNICs are used in the data plane path. There are vNICs that are used for management access, OAM, and keep alive mechanisms, for example. The **`ctxPerDev`** recommendation must be enabled for vNICs that are expected to process a high packet load rather than all vNICs in the VNFC.

Dedicating CPUs to System Threads

The **`sched.cpu.latencySensitivity`** configuration described in the *Dedicating CPUs to a VNF Component* section of this document assigns exclusive affinity between physical CPU cores and the vCPUs of a VNFC. As discussed in the *Understanding the Data Plane Path* section of the document, there are hypervisor processes responsible for delivering network traffic to a VNFC and delivering network traffic from the component back to the network.

With the VM setting **`sched.cpu.latencySensitivity.sysContexts`**, system threads (Tx and Rx) are assigned exclusive physical CPU cores. In VMware vCloud NFV 1.5, which uses VMware vSphere 6.0 U2, only the Tx thread is assigned exclusive physical CPU core using the **`sysContexts`** configuration. Starting with VMware vCloud NFV 2.0 with VMware vSphere 6.5, both Tx and Rx threads are assigned exclusive physical CPU cores using this command.

A prerequisite to this configuration is that **`sched.cpu.latencySensitivity`** is set to **High**. To set **`sched.cpu.latencySensitivity.sysContexts`** follow these steps:

1. In the vSphere Web Client go to **VM Settings**.
2. Choose the **Options** tab and select **Advanced General** > **Configuration Parameters**.
3. Configure **`sched.cpu.latencySensitivity.sysContexts`** with a numerical value.

The numerical value assigned to **`sched.cpu.latencySensitivity.sysContexts`** must equal the number of active threads for the component. For example, if one receive thread exists and three Tx threads have been set using the **`ctxPerDev`** command, the value set must be 4. In this example, 4 physical CPU cores must be available and unreserved to facilitate the benefits of these combined commands.

The use of the **`sysContexts`** configuration also requires some level of CPU capacity planning. A NUMA node where the VNFC resides must have enough non-exclusive cores to satisfy the value configured in **`sysContexts.`** If that is not the case, the sysContext behavior will not take effect.

Recommendations:

1. Use **`sched.cpu.latencySensitivity`** set to **High** to provide a VNFC with exclusive CPU resources.
2. As needed, assign additional Tx threads to virtual network interfaces using **`ethernetX.ctxPerDev`**.
3. To ensure that system threads receive exclusive CPU resources use **`sched.cpu.latencySensitivity.sysContexts`**.

Section Summary

Providing dedicated compute resources to networking processes requires careful planning. As often is the case with engineering decisions, benefits and drawbacks must be evaluated. One can significantly increase the throughput performance of a data plane intensive VNF if the correct amount of resources are assigned. Yet, providing excessive resources without proper justification only ensures that compute resources are not available to the components or processes that need them, while the component reserving the resources gains no benefit from them. The recommended testing approach to help VNF vendors evaluate the benefits of the configurations outlined in this section can be found in *Appendix B: Lab Testing Advice*.

NFVI Design and Considerations to Support Data Plane Intensive Workloads

A typical VNF deployment includes three parties:

1. **The Communications Service Provider (CSP)**. The CSP is responsible for operating the virtual infrastructure as well as creating, maintaining, and monetizing the service.
2. **The VNF Vendor**. The VNF vendor is responsible for creating and packaging the VNF, continuing to develop it, and providing VNF life cycle support.
3. **The NFVI Environment Vendor**. The NFVI environment vendor is responsible for creating the components used to virtualize the compute, storage and networking, and operational tools required to manage the environment and consume the virtual resources.

The previous sections of this document focused on the methods a VNF vendor uses to optimize the VNF and its components for data plane performance. VMware provides extensive configuration options and guidance to accomplish this. Such activities are likely to be performed in the VNF vendor's own lab and will result in a packaged VNF and a deployment guide that provides mapping between telecommunications workload characteristics, for example the number of subscribers or throughput and the amount of resources required to support them.

The next section of this document describes the necessary approach the NFVI operator must take to ensure healthy data plane performance.

Host Preparation

To benefit from virtualization, common compute platforms are used. Since data plane workloads are demanding of CPU and network interfaces, newer CPUs with more cores and CPU cycles provide improved performance.

In addition to using the latest server technology, servers themselves must be setup to support data plane intensive VNFs. Default BIOS settings are not always configured for high performance, but are tuned for energy efficiency instead. Since the names of BIOS configurations differ between servers and BIOS manufacturers, generic terms are used here.

- **Power Management**. Set this to **high** or **maximum** to ensure that processor CPU cycles are always available and the processor does not step down its capabilities to preserve energy. On some systems, this option is called **High Performance Profile**.
- **Hyperthreading**. Enable this option in systems that support it. Some BIOSs refer to this as a **Logical Processor**. More information about hyperthreading is available from the VMware vSphere 6.5 Documentation Center.
- **Turbo Boost**. Enable this option. More information about Turbo Boost is available from the Frequently Asked Questions about Intel Turbo Boost Technology page of the Intel Web site.
- **NUMA Node Interleaving**. Be sure this option is disabled.

VMware, Inc. 3401 Hillview Avenue Palo Alto CA 94304 USA Tel 877-486-9273 Fax 650-427-5001 www.vmware.com

Network Interface Card Preparation

Another factor which contributes significantly to data plane performance is the NIC and its interaction with the physical and virtual environments. The choice of NIC could significantly affect performance.

These guidelines will assist you in preparing your virtualization environment for data plane intensive VNFs:

- When using VMware vCloud NFV 2.0 with Intel 10 Gigabit Ethernet NICs, activate the native ixgben driver to minimize the overhead of translating data structures between VMkernel and VMKLinux, improve throughput, and reduce latency. See the Knowledge Base article Enabling and Disabling Native Drivers in ESXi 6.5 (2147565) for more information.
- Ensure that your NIC drivers are up to date. NIC vendors contribute improved NIC drivers on an ongoing basis. As a result, later drivers than those shipped with your vSphere version are almost always available. The Knowledge Base article How to download and install async drivers in ESXi 5.x/6.x (2005205) provides instruction on how to identify and install driver updates.
- Drivers often support many features and protocols. Deactivating some features, such as Fiber Channel Over Ethernet (FCoE), will reduce occasional interruptions in driver operations, improving data plane performance. Starting with the VMware vCloud NFV 2.0 release, when the hosts use Intel NICs, no FCoE support is provided. See the Knowledge Base article End of Availability and End of Support for FCoE on Intel Network Controllers (2147786) for more information.
- Most modern high performance servers include several different PCI Express (PCIe) slots. These slots range in speed and use a naming scheme such as PCIe x8 and PCIe x16. Placing a single port 10 Gigabit Ethernet NIC in a PCIe x8 provides sufficient performance. Higher rate cards such as dual-port 10 GigabitEthernet, or 40 GigabitEthernet, will benefit from the use of the PCIe x16 slots.

VMware vSphere Preparation for Data Plane Intensive VNFs

The configuration described to this point in the document relates to the server and the NICs. The hypervisor, ESXi, is the layer abstracting the physical resources for many VMs to benefit from the same physical components. There are several ESXi configuration paradigms that will benefit data plane intensive workloads. In servers intended for use by data plane intensive workloads, the following implementation is recommended.

Buffer Size Alignment

There are several buffers that hold network traffic as it flows between the ingress physical NIC and the egress NIC, out of the server. Proper alignment of the user-configurable buffers along this path will improve data plane throughput performance. The general guidance for buffer size as described below, specifies that a receiving buffer must be somewhat smaller than a transmitting buffer.

In addition, VNFCs also use buffers in the application space. The same guidelines regarding buffers applies here: the receive buffer must be configured with a smaller value than the transmit buffer. When possible, these values must also consider the physical network interface buffers, such that the receive buffer size on the physical NIC has a smaller value than the VNFC receive buffer, and the largest value in the Tx chain is the size of the transmit buffer on the physical NIC.

SplitTx Mode

SplitTx mode is a new capability introduced in VMware vSphere® 6.5. SplitTx is configured for a pNIC on a host. When active, it enables the hypervisor to use two transmission threads for the pipeline, one for the vNIC backend, another for the pNIC. The effect of this configuration is that VNFCs residing on the host and using the vNIC connected to the pNIC will benefit from increased performance for packets egressing the

VNF. Since the configuration is host based, this command line must be used to activate this feature for a specific vmnic: **`vsish –e set /net/pNics/vmnicX/sched/txMode 1`**. **`X`** must be replaced with the network interface number.

NSX for vSphere Usage and Configuration

VMware NSX® for vSphere® is the virtual networking component used in the vCloud NFV solution. NSX for vSphere enables the creation of entire networks in software and embeds them in the hypervisor layer, abstracted from the underlying physical hardware. There are several components that comprise NSX for vSphere, including firewalls, routers, and load balancers. Careful consideration must be given to the placement of the NSX for vSphere components in the virtual infrastructure, and to NSX for vSphere use by the VNFs and their components. Guidance regarding NSX for vSphere components is provided in this section of the document. For further information about NSX for vSphere, review the NSX Administration Guide for NSX for vSphere 6.2, or the NSX Administration Guide for NSX for vSphere 6.3 for the VMware vCloud NFV 2.0 user.

Distributed Firewall

The NSX for vSphere distributed firewall (DFW) focuses on East-West access controls. Since the DFW components are implemented using a hypervisor kernel module, this type of firewall is the one most likely to affect the data plane performance of a VNF component. The efficiency and utility of using a software firewall in each virtual host must be weighed against the data plane requirements of the components. When traffic control between VNF components is not required, configure the NSX firewall to exclude the VNF components from its firewall list. The NSX firewall configuration allows the administrator to exclude a VM from firewall protection. See the NSX Administration Guide for NSX for vSphere 6.2, or the NSX Administration Guide for NSX for vSphere 6.3 for a detailed explanation of the procedure to exclude a virtual machine from firewall protection.

Edge Services Gateway

The Edge Services Gateway (ESG) is a virtual appliance that implements services such as firewall, DHCP, network address translation (NAT), virtual private networking (VPN), load balancing, and routing. In the vCloud NFV reference architecture, instances of NSX ESGs are used to facilitate the routing of network traffic from the physical network to the virtual network, and to provide logical delineation for network services.

The NSX ESG behaves like any other service rich router. Each packet is processed based on the active service, and routing lookup is performed. As such, the NSX ESG can become a data plane bottle neck. The ESG services, such as the NAT and firewalls must be deactivated to improve performance, leaving only routing as the active service. Under a high traffic load the ESG can become a performance limiting factor. In this case, the data plane intensive VNFs must bypass the NSX ESGs and use top-of-rack switches or routers to get data in and out of the virtual domain.

Section Summary

As the telecommunications industry transitions from physical devices to VNFs, more and more advances in virtualization and server hardware are taking place. As with every technological transition, requirements and development converge over time. VMware is committed to continuing the development of the most widely-used virtualization platform in the world, to meet the requirements of CSP customers. As this white paper demonstrates, there are many possible ways to improve the performance of data plane intensive VNFs. Working closely with VNF vendors, the vCloud NFV team has developed significant performance improvements across various use cases. The guidance in this document will allow you to optimize the performance of your VNF or NFV infrastructure.

Appendix A: Summary Recommendations

Appendix A of this white paper provides configuration information, including that discussed within the body of the document and the information provided within some of the additional resources included here. None of the configuration parameters described should be applied blindly, but must first be tested in a lab while measuring the effect on the environment and the VNF performance.

BIOS Configuration

Different BIOS manufacturers use different names for their BIOS functions. This document attempts to reflect general naming conventions where possible. Consult your server documentation for exact configuration details.

PARAMETER	VALUE	CONFIGURATION
Power Management	Maximum Performance / High Performance	Not relevant
Hyperthreading	Enable	Not relevant
Turbo Boost	Enable	Not relevant
C States	Disable	Not relevant
Intel VT Technology	Enable	Not relevant
QPI Power Management	Disable	Not relevant
Execute Disable Bit	Enable	Not relevant
Node Interleaving	Disable	Not relevant

Hypervisor

PARAMETER	VALUE	CONFIGURATION
NIC Driver	ixgbe or ixgben for VMware vSphere 6.5	Not relevant
FCoE Configuration	Disable	esxcfg-module ixgbe -s CNA=0,0
Rx Descriptor	2048 (example)	ethtool -G $port1 rx 2048 ethtool -G $port2 rx 2048
Tx Descriptor	4096 (example)	ethtool -G $port1 tx 4096 ethtool -G $port2 tx 4096
Software Tx Queue Descriptor	6000	esxcli system settings advanced set -i 6000 -o /Net/MaxNetifTxQueueLen
TxSplit Mode	Activate	vsish –e set /net/pNics/vmnicX/sched/txMode 1

Virtual Machine (VNFC) Configuration

PARAMETER	VALUE	CONFIGURATION
VIRTUAL CPU		
Shares	High	VM Settings > Resources > CPU > CPU Shares set to High
Reservation	Maximum per Number of vCPU Allocated to VM	VM Settings > Resources > CPU > Reservation set to number of vCPUs multiplied by processor base frequency
Limit	Unlimited	VM Settings > Resources > CPU > Limit set to Unlimited
MEMORY		
Shares	High	VM Settings > Resources > Memory > Shares set to High
Reservation	Maximum Necessary	VM Settings > Resources > Memory > Reservation set to Maximum needed
Limit	Unlimited	VM Settings > Resources > Memory > Limit set to Unlimited
OTHER CONFIGURATION		
Hardware Version	11 for vSphere 6.0 U2 13 for vSphere 6.5	Right-Click on the VM in vCenter Server and choose Upgrade Hardware Version
Virtual NIC	VMXNET3	Set Adapter Type to VMXNET3
NUMA Affinity	Align to physical NIC NUMA	VM Settings > Options > Advanced General > Configuration Parameters `numa.nodeAffinity = "0, 1, …"`
sysContexts	Set to Number of Threads Pinned	VM Settings > Options > Advanced General > Configuration Parameters `sched.cpu.latencySensitivity.sysContexts = "Number of threads to be pinned"`
LatencySensitivity	Set to High	VM Settings > Options > Advanced General > Configuration Parameters `sched.cpu.latencySensitivity = "high"`
Tx Thread Allocation	For all vNIC Transmitting Traffic	VM Settings > Options > Advanced General > Configuration Parameters `ethernetX.ctxPerDev = "1"` where `X` is replaced with the vmnic number

VMware, Inc. 3401 Hillview Avenue Palo Alto CA 94304 USA Tel 877-486-9273 Fax 650-427-5001 www.vmware.com

Appendix B: Lab Testing Advice

Testing the performance of VNFs on NFV infrastructure is the subject of a growing body of work by organizations such as the IETF Benchmarking Working Group. For more information read the Considerations for Benchmarking Virtual Network Functions and Their Infrastructure paper.

Since CSPs are bound by strict regulation and often transport mission critical data, CSPs expect that the software and hardware solutions they use will meet certain key performance indicators. Historically, most network functions deployed in a CSP network have been rigorously tested by their supplier, and often by the CSP. This practice is not expected to stop when network functions transition from hardware to software. Vendors creating VNFs are expected to collaborate with NFVI suppliers to understand the performance the VNFs can achieve. Collaboration between VNF and infrastructure vendors will ensure that CSPs receive efficient and high-performing virtual network solutions.

Lab testing prior to deployment is the key to successfully deploying data plane intensive VNFs. The goals of these tests are to:

- Understand the physical resources required by the VNF components to perform optimally.
- Capture the VNF resource requirements in a deployment guide to help the CSP asses the number of hosts and CPUs, and the amount of memory and storage, required to deploy a certain size of VNF.
- Provide guidance to the CSP on how to configure the underlying NFVI.
- Incorporate configurations into the VNF component descriptor.

Testing Methodology

The general testing methodology recommended includes these steps:

1. Start from a baseline, non-optimized configuration and measure the achieved packet rate and throughput through the system. See the Benchmarking Terminology for Network Interconnection Devices memo RFC 1242 from the IETF Network Working Group for terminology definitions.
    - As much as possible, use the simplified tester configuration of a single packet size. See the Benchmarking Methodology for Network Interconnect Devices memo RFC 2544 from the IETF Network Working Group for packet size recommendations. While the smaller packet sizes, such as 40 bytes, are unrealistic in production networks they do allow stressing the system to its limits.
    - Use UDP packets for your tests. Using TCP for throughput tests is not recommended since TCP will back off its transmission as soon as packet loss is seen, thereby reducing the stress in the system.
2. The server type, NIC model, and CPUs must be kept constant and reported with test results. This information must also be provided in any deployment guide document as it greatly affects the validity of any recommendation.
3. A test case must explore one and only one vector. When attempting to identify the highest performing configuration for a VNF, focus on changing one configuration parameter at a time and repeating the exact same tester configuration with each additional test run.

vmware

VMware, Inc. 3401 Hillview Avenue Palo Alto CA 94304 USA Tel 877-486-9273 Fax 650-427-5001 www.vmware.com

4. Test each VNFC separately. There are two reasons for this recommendation:
    - Identifying and solving a bottle neck is more efficient when there is only one software component being tested.
    - Maximizing the performance of a component will inevitably identify the number of resources required for the component. In some cases where the VNF has several components, each requiring physical resources to reach optimal configuration, the placement of VNF components can be influenced during the tests. For example, a VNF component responsible for load balancing may require a whole host worth of resources to classify and load balance traffic before the traffic reaches the data processing VNF component.
5. Packet loss rate must be considered and reported. Depending on the VNF type and the standards requirements, some packet loss is likely to be acceptable. For example, the Digital cellular telecommunications system (Phase 2+) (GSM); Universal Mobile Telecommunications System (UMTS); LTE; Policy and charging control architecture (ETSI TS 123 203) technical specification defines 10^{-2} packet loss rate as acceptable for conversational voice and 10^{-3} packet loss rate as acceptable for live streaming video. A packet loss ratio must therefore be set according to the nature of the VNF.
6. Packet loss rate must be measured and correlated with the CPU load on the virtual infrastructure. If CPUs are clearly the bottle neck, for example if CPUs report 100% utilization during the whole test, adding CPU resources and using the methods described in this white paper are appropriate.
7. Once each component has been optimized and the VNF architecture is set, repeating the tests with the entire VNF will be of value. In this case, using a mix of packet sizes, referred to as an Internet mix (IMIX), is prudent. This identifies how the VNF is likely to behave under real world conditions. The packet breakdown of the IMIX must be defined by the VNF type.

VMware, Inc. 3401 Hillview Avenue Palo Alto CA 94304 USA Tel 877-486-9273 Fax 650-427-5001 www.vmware.com

Author and Contributors

Jambi Ganbar, Technical Solutions Manager, NFV Solution Engineering, VMware wrote this paper.

Many thanks for contributions from:

- Danny Lin, Senior Director, NFV Solutions Engineering, VMware
- Mahesh Pandurangan, Staff Engineer, NFV Solutions Engineering, VMware
- Ramachandran Sathyanarayanan, Solution Consultant, NFV Solutions Engineering, VMware
- Piyush Kumar Singh, Solution Consultant, NFV Solutions Engineering, VMware
- Bob Goldsand, Staff Global Solutions Architect, Solution Engineering, VMware
- Christian Prediger, Consulting Architect, EMEA Consulting, VMware
- Jin Heo, Staff Engineer, Research and Development, VMware
- Rishi Mehta, Staff Engineer, Research and Development, VMware
- Richard Lu, Senior Staff Engineer, Research and Development, VMware
- Aditya Holla, Staff Engineer, Research and Development, VMware
- Indranil Bal, Solution Consultant, NFV Solutions Engineering, VMware

VMware, Inc. 3401 Hillview Avenue Palo Alto CA 94304 USA Tel 877-486-9273 Fax 650-427-5001 www.vmware.com