NSX-T Networking Best Practices

NSX-T 1.1

This document supports the version of each product listed and supports all subsequent versions until the document is replaced by a new edition. To check for more recent editions of this document, see http://www.vmware.com/support/pubs.

vmware

You can find the most up-to-date technical documentation on the VMware Web site at:
http://www.vmware.com/support/
The VMware Web site also provides the latest product updates.
If you have comments about this documentation, submit your feedback to:
docfeedback@vmware.com

VMware, Inc.
3401 Hillview Ave.
Palo Alto, CA 94304
www.vmware.com

Contents

NSX-T Networking Best Practices

The *NSX-T Networking Best Practices Guide*, provides best practice recommendations and configuration instructions to enhance NSX-T performance.

Intended Audience

This information is intended for anyone who wants to maximize the NSX-T network performance. This information is written for experienced system administrators who are familiar with virtual machine technology, virtual networking, and virtual datacenter operations.

This manual assumes familiarity with a virtual machine management service—such as VMware vSphere 6.5, 6.0 or 5.5, including VMware ESX, vCenter Server, and the vSphere Web Client, VMware Tools—or another virtual machine management service with kernel-based virtual machines (KVMs).

VMware Technical Publications Glossary

VMware Technical Publications provides a glossary of terms that might be unfamiliar to you. For definitions of terms as they are used in VMware technical documentation, go to http://www.vmware.com/support/pubs.

1 Hardware Recommendations

Before you undertake network optimization effort, you must understand the physical aspects of the network. Consider implementing the following hardware on your physical layout:

- Server-class network interface cards (NICs) for best performance
- Latest drivers for NICs
- Latest firmware version for ESX
- Network infrastructure between the source and destination NICs should not introduce bottlenecks.
  For example, if both NICs are 10Gb/s, all the cables and switches must be of the same speed and the switches must not be configured to a lower speed.

NIC Considerations

For the best network performance, use NICs that support specific hardware features and that support offloads of encapsulated packets.

To find out supported offloads for an adapter (IPv4 and IPv6), see the product specification or datasheet provided by the adapter vendor, or refer to the IO compatibility guide [4].

Supported NIC Features

- Checksum offload (CSO)
- TCP segmentation offload (TSO)
- UDP Checksum offload (CSO)
- Encapsulated Packets (Overlay)
  - Geneve TCP/UDP Checksum offload (CSO)
  - Geneve TCP segmentation offload (TSO)
  - Geneve Rx filter
- Geneve OAM packet filter
- Ability to handle high-memory DMA (64-bit DMA addresses)
- Ability to handle multiple Scatter Gather elements per Tx frame
- Jumbo frames (JF)
- Receive Side Scaling (RSS)

Network cards must be installed in slots with enough bandwidth to support their maximum throughput.

PCI Slots	NIC
PCIe x8 or later	Single-port 10Gb/s
PCI-X 266	Single-port 10Gb/s
PCIe x16 or later	Dual-port 10Gb/s
PCIe 2.0	Number of lanes (x8 or x16) can be reduced accordingly. There should be no bridge chip in the path to the actual Ethernet device such as, PCI-X to PCIe or PCIe to PCI-X. Determine that there are no embedded bridge chips on the device which can reduce performance.
PCI Gen3 x8 slots or later PCI Gen3 x16 slots or later	40Gb/s 100Gb/s

Multiple physical NICs between a virtual switch (vSwitch) and the physical network constitute a NIC team. NIC teams can provide passive failover in the event of hardware failure or network outage. In some configurations, NIC teams can increase performance by distributing the traffic across those physical NICs.

When you use load balancing across multiple physical NICs connected to one vSwitch, all the NICs should have the same line speed.

If the physical network switch to which your physical NICs are connected supports Link Aggregation Control Protocol (LACP), configuring both the physical network switches and the vSwitch to use LACP can increase throughput and availability.

MTU Size

To support overlay traffic, NSX-T supports the MTU size of 1600B (1500B with 100 extra bytes for encapsulation) or larger on all the components. Recommended minimum MTU size is 1600B.

Dedicated Host for Bridge

A bridge provides connectivity between overlay traffic and VLAN traffic. It is recommended to use a dedicated host if a bridge is configured on that host since the workload can be high.

BIOS Settings

The default hardware BIOS settings on servers might not always be the best choice for optimal performance. When you configure a new server check the BIOS settings.

- Run the latest version of the BIOS available for your system.
- BIOS must enable all populated processor sockets and all the cores in each socket.
- Enable the Turbo Boost in the BIOS if your processors support it.
- Select High Performance Profile in the BIOS setting for CPU to allow maximum CPU performance. Note that this might increase power consumption.
- Some NUMA-capable systems provide an option in the BIOS to disable NUMA by enabling node interleaving. In most cases, disabling the node interleaving and leaving NUMA enabled is better for performance.
- Enable hardware-assisted virtualization features such as, VT-x, AMD-V, EPT, RVI in the BIOS.
- If the BIOS allows the memory scrubbing rate to be configured, keep the manufacturer default setting.

2 ESXi Networking Recommendations

Consider the guidance provided for ESXi network optimization.

vSphere Version Recommendation

For best performance with NSX-T, use vSphere 6.5 which includes the following key features:

- Uplink SW LRO support has a significantly improved performance for firewall and available by default.
  This feature can be used with NICs that support NetQueue with VLAN Traffic and the NICs that support Geneve Rx filter for Geneve Traffic. Uplink SW LRO aggregate packets as soon as the physical NIC receives it to reduce the number of packets processed by the networking stack.

- ESXi pNIC offload support for Geneve is enhanced in the vSphere 6.5 release.

Features	vSphere 6.0 and Update releases	vSphere 6.5 (2016)
Inner frame TCP/UDP CSO/TSO and software emulation	Yes (*)	Yes
Outer UDP checksum offload and software emulation	No	Yes
Geneve Rx filter	No	Yes
Geneve OAM queue for management traffic	No	Yes

(*) The TSO support in vSphere 6.0 has some limitation. If NICs have constraints in number of SG elements in DMA engine such as, i40e driver, maximum L2 payload size and maximum MSS size, this combination is not supported in vSphere 6.0.

- Support of multiple Tx worlds which improves packet transmission performance by using more worlds to process packets from a particular vNIC queue.
- Split Rx mode will have a dedicated kernel thread for Receive Processing in vSphere 6.5. In the prior vSphere releases, SplitRx mode used to share the VM/VNIC Tx Processing thread for doing Receive processing for the VNIC.

Physical NIC Configuration

You can configure the supported physical NIC configurations for NSX-T to tune the performance.

Receive Side Scaling (RSS)

Receive Side Scaling (RSS) allows network packets from a NIC to be processed in parallel on multiple CPUs by creating a different thread for each hardware queue. ESX supports two variations of RSS - NetQueue and Device. When used in NetQueue mode which is default, ESX controls how traffic is distributed to multiple hardware queues. In Device RSS mode, hardware is in complete control of how traffic is distributed to multiple hardware queues.

NetQueue

NetQueue is a logical queue concept unique to ESXi that can be mapped to one or more hardware queues. By default, each logical queue maps to single hardware queue and there is a single thread processing that queue. NetQueue redirects traffic so that packets destined to a certain VM are distributed to the same NetQueue. NetQueue allows data interrupt processing to be affinitized to the CPU cores associated with individual VMs, improving receive-side networking performance by providing better NUMA locality.

Some pNIC drivers also support a feature called queue pairing. Queue pairing indicates to the ESXi uplink layer that the receive thread (NetPoll) also processes completion of transmitted packets on a paired transmit queue. This feature could even further improve the CPU efficiency. To disable queue pairing use the command,

```
# esxcli system settings advanced set -o
/Net/NetNetqRxQueueFeatPairEnable -i 0
```

In the NetQueue model, there is a default queue and single or multiple non-default queues. All Broadcast / Unknown unicast / Multicast (BUM) traffic is handled by the default NetQueue. Based on the load balance algorithm, traffic destined to a VM can be redirected to a non-default queue by pushing down the VM's MAC address as the filter to the physical queue on the NIC.

In the NSX overlay environment, the VM MAC address might not be seen by the NIC because it is encapsulated. There are some NICs that support Rx filter for encapsulated traffic (Geneve Rx filter) and can see inner MAC. In that case, NetQueue works well for encapsulated packets. However, when the NIC can only see the outer MAC address which is VTEP's MAC address, all overlay traffic destined to this VTEP is redirected to a single NetQueue. To improve the scalability for encapsulated traffic in the absence of Geneve Rx filter, NetQueue RSS was introduced.

NetQueue RSS

When NetQueue RSS is enabled, there is a special NetQueue created called NetQueue RSS, which is associated with up to four hardware queues. Overlay traffic can be redirected to the NetQueue RSS and further distributed to multiple hardware queues based on 5-tuple hash. While this might increase network throughput for a NIC that receives packets at a high rate, it can also increase CPU overhead since there is NUMA locality loss.

The command to enable NetQueue RSS is device specific, please refer to device driver manual for each device to enable it.

Default Queue RSS

All BUM Traffic is handled by the default queue. Default queue is backed by single hardware queue and is serviced by a single thread by default. This could result in performance degradation for certain hosts which received a lot of BUM traffic. To overcome this limitation, vSphere 6.0 or vSphere 6.5 later version (depending on driver type used) has introduced Default Queue RSS.

When Default Queue RSS enabled, default queue is serviced by multiple hardware queues. All the traffic it receives is distributed among different hardware queues which are serviced by different hypervisor threads.

Default Queue RSS is useful for high broadcast/multicast traffic or for appliances that process packets on multiple MAC addresses which are unknown to the hypervisor such as, VMs on promiscuous mode and other gateway appliances.

Enable Default Queue RSS

You can enable the Default Queue RSS for bnx2x NICs.

Procedure

1. Open a command prompt.
2. Enable Default Queue RSS by configuring the module load parameter.
   ```
   esxcfg-module -s "rss_on_default_queue=1
   num_queues_on_default_queue=4" bnx2x
   ```

Device Mode RSS

vSphere 6.0 and vSphere 6.5 depending on the driver type has support for Device Mode RSS, which is an RSS feature on NIC. For the Device Mode RSS, the device controls all the hardware queues. A NIC uses the RSS engine to drive traffic among multiple queues.
The Device Mode RSS is recommended when you configure bridge or Edge on the transport node. The Device Mode RSS has two disadvantages.

- All queues are used irrespective of traffic load. When there is a low network traffic, this traffic that could be serviced by fewer queues is distributed among all queues

instead. This results in a higher interrupt rate as each queue generates its own interrupts.

- The unicast traffic for one VM can be serviced by multiple queues, which results in loss of locality especially for NUMA. VMs are placed on the same NUMA node where the thread servicing the traffic is placed. With Device Mode RSS, all VMs share all queues and therefore, cannot get proper isolation.

The command to enable NetQueue RSS is device specific, please refer to device driver manual for each device to enable it.

RSS Recommendation Summary

NetQueue non-RSS mode is enabled by default. All other features must be enabled explicitly.

		VLAN Traffic	**Overlay Traffic**	**Bridging Traffic**
NetQueue		Yes for unicast traffic	Yes, if Geneve RX filter is supported	
NetQueue Mode RSS	NetQueue RSS		Yes, if Geneve RX filter is NOT supported	
	Default queue RSS	Yes for BUM traffic		Yes
Device Mode RSS				Yes, if default queue RSS is NOT available

Queue Size

The ESXi uplink pNIC layer also maintains a software Tx or Rx queue of packets queued for transmission or reception, which by default holds 500 packets. If the workload is I/O intensive with large bursts of packets, this queue may overflow leading to packets being dropped in the uplink layer.

Increase Queue Size

You can increase the queue size up to 10,000 packets.

Depending on the physical NIC and the specific version of the ESXi driver being used on the ESXi host, packets can be dropped in the pNIC driver because the transmit or reception ring on the pNIC is too small and is full.

Procedure

- Increase the queue size for Tx or Rx.

```
# esxcli system settings advanced set -i 10000 -o
/Net/MaxNetifTxQueueLen
```

- Increase the size of the ring in the pNIC drivers for Tx.

```
# ethtool -G vmnic0 tx 4096
```

  This command increases the Tx ring size to 4096 entries.

- Determine the maximum size you can set for a specific pNIC driver and current Tx ring.

```
# ethtool -g vmnic0

Ring parameters for vmnic0:
Pre-set maximums:
RX: 4096 RX Mini: 0
RX Jumbo: 0
TX: 4096
Current hardware settings:
RX: 512
RX Mini: 0
RX Jumbo: 0
TX: 4096
```

Virtual NIC Configurations

For the best virtual NIC performance, use the VMXNET3 paravirtualized network adapter for the supported operating systems. The virtual machine must use virtual hardware version 7 or later and, in some cases, VMware Tools be installed in the guest operating system.

Interrupt Coalescing

Virtual network interrupt coalescing can reduce the number of interrupts posted to VM to decrease CPU utilization. The reduction in virtual networking overhead might allow more virtual machines to run on a single ESXi host. Virtual network interrupt coalescing might increase network latency from a few hundred microseconds to a few milliseconds. Many workloads are not impacted by the additional network latency.

Virtual network interrupt coalescing is enabled for all virtual NICs in ESXi by default. For VMXNET3 virtual NICs, you can be disable or set to a static value by changing the ethernetX.coalescingScheme variable.

Disabling virtual interrupt coalescing increases interrupts to increase CPU utilization. It might also lower network latency.

Setting virtual network interrupt coalescing to a static value causes ESXi to queue a predefined number of packets before interrupting the virtual machine or transmitting the packets. When set to static, ESXi queues up to 64 packets by default. This value can be changed between one and 64 in the ethernetX.coalescingParams variable.

Increasing the queue size can reduce the number of context switches between the virtual machine and the VMkernel, reducing CPU utilization in the virtual machine and VMkernel.

ESXi waits for a maximum of four milliseconds before sending an interrupt or transmitting the packets. Other events, such as idle virtual machine might also trigger virtual machine interrupts or packet transmission, so packets are rarely delayed the entire four milliseconds.

Disable Virtual Interrupt Coalescing

You can disable virtual interrupt coalescing for VMXNET3 virtual NIC or change the scheme and parameters from the vSphere Client.

Procedure

1. Log in to the vSphere Client.
2. Select the virtual machine to configure.
3. Click **Edit virtual machine settings**.
4. Under the **Options** tab, select **General** > **Configuration Parameters**.
5. Locate the ethernetX.coalescingScheme variable.
   If the variable is not available, click **Add Row** and type `ethernetX.coalescingScheme`.
6. Disable the ethernetX.coalescingScheme variable to disable virtual interrupt coalescing.
   Where X is the number of the NIC.
7. Set the ethernetX.coalescingScheme variable to desired scheme name.
   For example, set the name to rbc for rate based coalescing.
   Where X is the number of the NIC.
8. Change the value of ethernetX.coalescingParams to desired value.
   For example, set the value to 16000 for rbc.
9. Restart the virtual machine.

Receive Side Scaling (RSS)

VMXNET3 devices support multiple queues for many guest operating systems that natively support RSS. The supported guest operating systems include Windows Server 2003 SP2 or later, Windows 7 or later, and some Linux distributions.

VMXNET3 drivers in the vSphere 5.0 and later versions of VMware Tools and VMXNET3 drivers included with Linux kernel 2.6.37 and later have multiple receive queues enabled by default. See, KB article 2020567.

When multiple receive queues are enabled, RSS configures the virtual machine to receive from 1, 2, 4, or 8 queues. It will choose the largest of these values that is less than or equal to the number of vCPUs in that virtual machine.

On Windows, the default number of transmit queues is one regardless of the number of receive queues. If the driver is configured to use multiple transmit queues, the number of transmit queues is the same as the number of receive queues.

On Linux, the default number of transmit queues is the same as the number of receive queues.

To obtain maximum performance for your workloads and resource availability you can add different values for the receive queues, the value must be set to a power of two and can be a maximum of 8 or the number of vCPUs, whichever is lower. You can also configure the driver to use multiple transmit queues. With the exception in Linux, where the number of transmit queues matches the number of receive queues. You can modify the settings in the advanced driver configuration tab within the guest operating system.

Queue Size

Overflowing receive queue in the receiver network device can cause low receive throughput in a virtual machine. If the receive buffers in the guest operating system network driver are insufficient, packets are dropped in the VMkernel. The dropped packets degrade the network throughput. You can increase the number of the receive buffers, which might increase the host physical CPU workload.

For VMXNET3, the default value of the receive and transmit buffers is controlled by the guest driver. The maximum possible value for the receive and transmit buffers is 4096. To modify the default values within the guest, you can use the ethtool utility in Linux and configure the Device Properties in Windows. See, KB article 1010071.

SplitRx Mode

SplitRx mode uses multiple physical CPUs to process network packets received in a single network queue to significantly improve network performance for certain workloads. These workloads include:

- For multiple virtual machines on an ESXi host receiving multicast traffic from the same source, the SplitRx mode improves throughput and CPU efficiency for these workloads.
- For traffic using the vNetwork Appliance or DVFilter API between two virtual machines on the same ESXi host, the SplitRx mode improves throughput and maximum packet rates for these workloads.

In vSphere 5.1 and later, SplitRx mode is enabled for the VMXNET3 virtual NIC, which is the only adapter type it is supported. The ESXi host detects the network queue on a physical NIC that is heavily utilized and getting more than 10,000 broadcast/multicast packets per second.

Disable SplitRx Mode for an ESXi Host

For better CPU efficiency, for example when the packet rate received from the queue is low the SplitRx mode can be disabled for an ESXi host from the vSphere Client.

Procedure

1. Select the ESXi host to disable.
2. Under the **Configuration** tab, in the **Software** pane, click **Advanced Settings**.
3. Locate the **NetSplitRxMode** variable.
4. Click the value.
5. Set the NetSplitRxMode value to `0` to disable the SplitRx mode for the ESXi host.
   The default value is 1, which enables the SplitRx mode for the ESXi host.
   This change is immediate and does not require the ESXi host to be restarted.

Enable or Disable SplitRx Mode for a Virtual NIC

The SplitRx mode can be configured on each virtual NIC to override the global NetSplitRxMode variable setting from the vSphere client.

Procedure

1. Select the virtual machine to configure.
2. Click **Edit virtual machine settings**.
3. Under the **Options** tab, select **General** > **Configuration Parameters**.
4. Locate the **ethernetX.emuRxMode** variable.
   Where X is the number of the virtual NIC. If the variable is not available, click **Add Row** and type `ethernetX.emuRxMode`.
5. Click the value.
6. Set the ethernetX.emuRxMode value to `0` to disable the SplitRx mode for ethernetX.
   The default value is 1, which enables the SplitRx mode for ethernetX.
7. Save the changes.
   - Restart the virtual machine.
   - Disconnect and reconnect the virtual NIC from the virtual machine.

Multiple Tx Worlds

vSphere by default configures 1 helper world for a VM to handle transmit traffic which is sufficient for most workloads. For certain workloads which require high packet rate transmits such as NSX Edge, vSphere has a special configuration to enable either 1 helper world per vNIC or 1 helper world per vNIC queue. vSphere 6.5 supports the 1 helper world per vNIC queue. vSphere 5.0 and later support the helper world per vNIC option.

Change Tx World Value

When net-stats show the Tx world runs at 100% for a VM, you might want to change this option to use more Tx worlds.

Procedure

1. Navigate to the vsi node /config/Net/intOpts/NetVMTxType.
2. Set the node value to 2 to configure a helper world for each vNIC Tx queue.

Performance Tuning for Different Workloads

Consider performance guidance for latency sensitive, Telco, and NFV workloads.

Latency Sensitive Workloads

To support virtual machines with strict latency requirements, vSphere 5.5 has a per-VM feature called Latency Sensitivity. Among other optimizations, this feature allows virtual machines to exclusively own physical cores to avoid overhead related to CPU scheduling and contention.

Combined with a passthrough functionality, which bypasses the network virtualization layer, applications can achieve near-native performance in both response time and jitter. [1]

Telco and NFV Workloads

The vSphere ESXi hypervisor provides a high-performance and competitive platform that effectively runs many Tier 1 application workloads in virtual machines. By default, ESXi has been heavily tuned for driving high I/O throughput efficiently by utilizing fewer CPU cycles and conserving power, as required by a wide range of workloads.

Telco and NFV application workloads are different from the typical Tier I enterprise application workloads, they tend to be any combination of latency sensitive, jitter sensitive, or demanding high packet rate throughputs or aggregate bandwidth, and need to be tuned for best performance on vSphere ESXi. [2]

To see how the feature configuration recommendations for physical NICs included in this section affect performance, please refer to [3].

3 KVM Networking Recommendations

Consider the guidance provided for KVM network optimization.

Linux Version Recommendation

Use the NSX-T supported Linux distribution, to take advantage of the performance improvements.

Check to see if your kernel, driver, and NIC support networking capabilities such as tunnel offloads and RSS.

Linux Hypervisor Connection Tracking

Distributed firewall (DFW) uses Linux conntrack for connection tracking. For workloads that require a high connection rate, where new connections are setup and removed rapidly over a sustained period, it is possible that the conntrack table gets full.

When the table is full conntrack generates the warning message, `nf_conntrack: table full, dropping packet` in the system log.

Procedure

- Set the variable timeout value.
  ```
  sysctl -w
  net.netfilter.nf_conntrack_tcp_timeout_time_wait=0
  ```

Physical NIC Configuration

To configure your NIC device for high performance, refer to the recommendations of your NIC vendor.

The details depend on the particular NIC device. Typical performance recommendations include RSS hashing configuration, number of queues, size of queues, interrupt coalescing, etc.

Handling Network Interrupts from Physical NICs

High-performance NICs support multiple queues for packet handling. Network traffic is hashed into multiple queues to be processed by multiple CPU cores concurrently.

A Linux kernel process called irqbalance maps interrupt requests (IRQ) associated with the NIC queues to the available CPU cores. For network intensive workloads you can modify the default behavior and map device IRQs to specific CPU cores.

To stop the irqbalance service, use the `service stop irqbalance` command.

To set affinity of the NIC device IRQs to specific CPU cores, check if your NIC driver has an available script. Use the command, `./<path-to-NIC-driver-scripts>/set_irq_affinity.sh <devname>` to run the script.

To improve performance, you can set the device IRQ affinity so that NIC interrupts are handled by CPU cores on the same NUMA node as the device. Use the command, `cat /sys/class/net/<devname>/device/numa_node` to find the NUMA node of the device.

Virtual NIC Configuration

Consider the virtual NIC configuration guidance for better performance.

Enable Multi-Queue Virtio for Virtual NICs

Multi-queue virtio-net provides enhanced network performance for a guest VM by allowing packet-processing ability to scale with the number of vCPUs.

Prerequisite

Verify that your operating system has support for multi-queue virtio.

Procedure

1. Open the guest XML configuration file.
2. Add XML code.
   ```
   <interface type='....'>
        <model type='virtio'/>
        <driver name='vhost' queues='N'/>
   </interface>
   ```
   Where N is the number of vCPUs assigned to the guest VM.
3. In the virtual machine, enable multi-queue virtio support.
   ```
   # ethtool -L ethX combined N
   ```

Queue Size

You can improve the throughput by increasing the txqueuelen value of the interface when the interface is connected to a high-speed network link.

You can experiment with modifying the default txqueuelen value to see if performance improves for your workload. For example, `#ifconfig ethX txqueuelen 10000` sets the txqueuelen value to 10000. Using a large txqueuelen value might increase the end-to-end packet latency.

4 Management Plane and Control Cluster Plane Recommendations

The Management Plane and Control Cluster Plane appliances must have sufficient CPU, memory, and storage resources for capacity and performance for your deployment size.
See the NSX-T Installation guide.

The NSX-T performance is dependent on the number of managed entities such as hosts and virtual machines. Exceeding the maximum specified in the NSX-T Configuration Maximums, is not supported and impacts performance.

Since the Management Plane and the Control Plane Appliances are distributed, as a best practice run them on different rack servers but refrain from having high network latency between them.

Logical Switch and DFW Rules

The Management Plane and the Control Plane aggregate before communicating between each other.

- As a best practice, configure all the logical switches in parallel before configuring the ports. Logical switch realization is batched and can decrease the deployment time if all of the logical switches are realized before ports are connected.
- Distributed Firewall rules should be consolidated into sections and it is recommended to minimize the number of sections.

References

[1] Deploying Extremely Latency-Sensitive Applications in VMware vSphere 5.5

http://www.vmware.com/content/dam/digitalmarketing/vmware/en/pdf/techpaper/latency-sensitive-perf-vsphere55-white-paper.pdf

[2] Best Practices for Performance Tuning of Telco and NFV Workloads in vSphere

http://www.vmware.com/techpapers/2015/best-practices-for-performance-tuning-of-telco-and-10479.html

[3] Leveraging NIC Technology to Improve Network Performance in VMware vSphere

http://www.vmware.com/content/dam/digitalmarketing/vmware/en/pdf/whitepaper/vmware-vsphere-pnics-performance-white-paper.pdf

[4] VMware Compatibility Guide

https://www.vmware.com/resources/compatibility/search.php?deviceCategory=io