VMware NSX-T Data Center

Performance considerations on Intel platforms

Samuel Kommu
Author

Foreword by Bruce Davie

Warning & Disclaimer

.

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

Table of Contents

List of Figures

List of Tables

About the Author

Samuel Kommu is a Sr. Technical Product Manager in the Networking and Security business unit at VMware. One of his focus areas is profiling network and compute resource usage of various applications and their performance, relevant bench-marking approaches, and design recommendations. Apart from performance, he focuses on helping customers migrate from VMware NSX for vSphere to VMware NSX-T Data Center and with enterprise design and architecture. He has over two decades of industry experience in diverse technology areas spanning application development, management services, system integration, performance profiling, enterprise design and architecture. Samuel is passionate about helping customers make educated decisions when designing enterprise data centers.

When away from work, Samuel loves to spend time with his wife Mini and their two boys Shalom and Nathaniel, on long drives, hikes and fishing trips exploring small historic towns, national parks and forests.

Acknowledgements

This booklet would not be possible, if not for the insights and extensive help and support from multiple individuals. I would like to thank the following people for their contribution to this booklet:

On the VMware front:

- Rishi Mehta, Peng Li, Boon Ang, Craige Jiang, Guolin Yang, Jayant Jain, and Yi Liao, for their invaluable insights into performance optimizations.
- Jochen Behrens and Ramakrishna Guduri, for their extensive work on Intel's QAT 8960s.
- Nghia Phan, Rosa Yu, Rick Correa, Ben Coleman, Anthony Dinh, JR Gasphar, Drew Sher, and Art Villamor, for going out of the way, over the years and specially in 2020, in getting labs setup and configured.
- Susan Wu and Scott Wieder, for help connecting different teams from two different companies for this initiative, and for Deb Howard, for her extensive editorial review and support.
- Devyani Pisolkar, for driving this initiative to fruition.
- Nimish Desai, for his support, as we explored the nuances of performance in the world of SDN.

On the Intel front:

- Karen Shemer, for quickly setting up multiple labs within Intel to benchmark and revalidate results.
- Shrikant M Shah, for insights into performance on Intel Platform.
- Saidulu Aldas, for insights into Intel's hardware and driver stack and for review.
- Pandi Maharajan, for helping with the technical review from Intel perspective.
- Shweta Jain, for providing hardware and multiple labs for testing and validation and for review.
- Victor Lee, for his support of the project concept.
- Marco Righini, for the support over the years exploring performance on Intel Platforms.
- Vinodhkumar Raghunathan, for help procuring the hardware for testing.

And finally, I would like to thank my wife Mini for her support and our kids, Shalom and Nathaniel, for the motivation.

Executive Summary

This eBook, "VMware NSX-T Performance Considerations on Intel Platforms" from VMware and Intel, offers guidance to architects, solution designers and networking specialists on how to tune and configure VMware NSX environments to maximize performance and achieve near line-rate throughput improvements. Showcasing the value of various key hardware-level features to drive and optimize software performance on x86 platforms, this report includes guidance on tools which can be leveraged to easily setup and benchmark performance. A summary of key findings can be applied to enhance your organization's VMware® NSX-T™ Data Center on Intel® platforms environment.

Foreword

For almost as long as computers have been connected to networks, performance considerations at the computer-to-network interface have been an area of focus for researchers and engineers. My first research project after finishing my Ph.D. focused on offloading a range of networking tasks from a server to an outboard packet processing engine. This was in 1989. Efforts to optimize the networking performance of servers have not stopped since.

Fast forward a few decades to 2011 when the team at Nicira shipped the first release of their “Network Virtualization Platform”. By this point it was common to use network interface cards (NICs) to offload some common networking tasks related to TCP processing, reducing the amount of time and CPU cycles spent in the server itself managing packets moving to and from the network. But as soon as you tried to encapsulate a packet with a non-TCP header, all these offloads stopped working, and performance dropped. The Nicira team, dependent on tunnel encapsulation to virtualize the network, came up with a novel solution called STT (Stateless Transport Tunneling). It was a hack in the best sense: repurposing the capabilities of existing NICs to make tunneling efficient, albeit in a way that wasn’t entirely standard.

Over the next few years, network virtualization became a core capability for modern data centers. As a result, the industry came together to define new tunneling encapsulations — first VXLAN and the Geneve — to achieve high performance for virtual overlays in a standardized way. And so today it is common to find NICs that support these standards and maintain efficient performance for a range of encapsulations.

If this little history tells you anything, it should be building data center networks to perform well and meet a range of other requirements (such as support for network virtualization overlays) is not a trivial task. It requires an understanding of a range of issues beyond simple speeds and feeds. This is where this book fills an important gap, as it tells you everything you need to know to achieve high performance for traffic passing between servers across virtual networks in modern data centers. With this concise book, you should be able to ensure you select the appropriate NICs for your data center and determine the best configuration of your components. Network virtualization provides a wide range of benefits ranging from better security to more agile network deployment; this book will help you ensure you also maintain peak performance.

Bruce Davie
VMware CTO
Asia Pacific & Japan

Typical Data Center Workloads

Workloads in the data center are typically TCP-based. Generally, 80% of the traffic flows within the data center are east/west, that is, communication between various compute nodes. The remaining 20% is north/south, that is, communication in and out of the data center. The following image (Figure 1: Data Center Traffic Pattern with NSX-T) shows the typical traffic flow distribution:

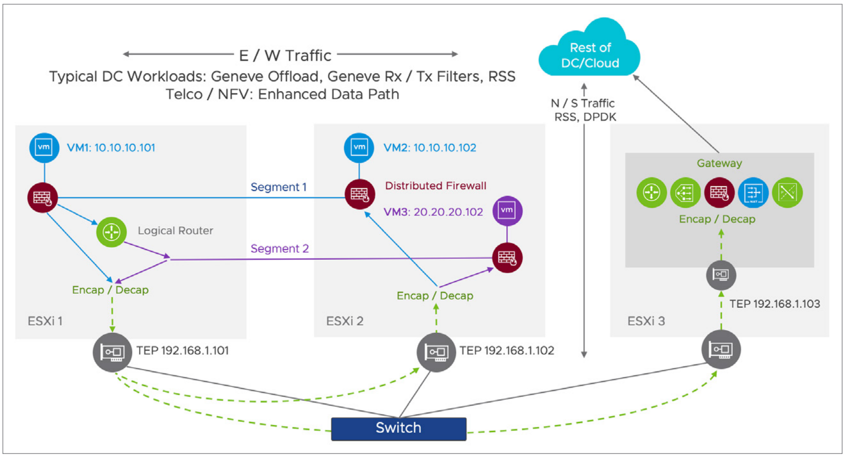


Figure 1 Data Center Traffic Pattern with NSX-T

This book primarily focuses on performance in terms of throughput for TCP-based workloads. There are some niche workloads such as NFV, where raw packet processing may be ideal, and the enhanced version of N-VDS called N-VDS (E) was designed to address these requirements. Check out the last part of this section for more details on N-VDS (E).

Next Generation Encapsulation – Geneve

Geneve, a draft RFC in the IETF standard body co-authored by VMware, Microsoft, Red Hat, and Intel, grew out of a need for an extensible protocol for network virtualization. With Geneve as the new framework for overlay/ network virtualization, understanding Geneve is foundational to rest of this eBook as the rest of the optimizations to be discussed revolve around Geneve.

With its options field length specified for each packet within the Geneve header, Geneve allows packing the header with arbitrary information into each packet. This flexibility offered by Geneve opens up doors for new use cases, as additional information may be embedded into the packet, to help track the packets path or for in depth packet flow analysis. For more information, please check out NSX-T Design Guide: https://communities.vmware.com/docs/DOC-37591.

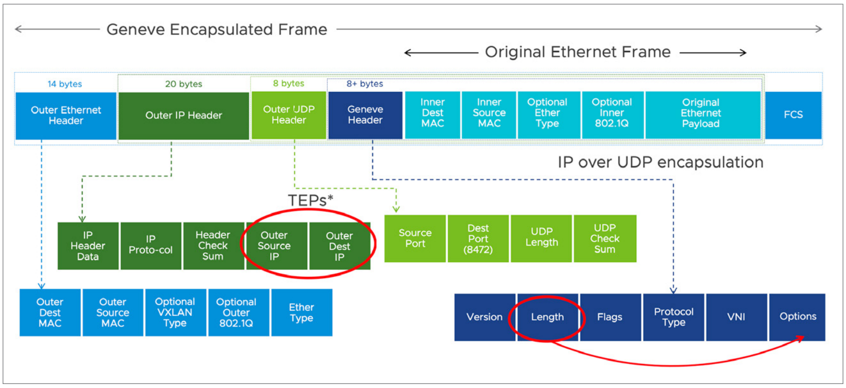


Figure 2 Geneve Header

The above image (Figure 2 Geneve Header) shows the location of the Length and Options fields within the Geneve header and also shows the location of TEP source and destination IP's.

For further insight into this topic, please check out the following blog post: https://octo.vmware.com/Geneve-vxlan-network-virtualization-encapsulations/

Geneve Offload

Geneve Ofload is simply TCP Segmentation Offload (TSO) tuned to understand Geneve headers. Since Geneve is not TCP traffic, NIC cards need to be aware of Geneve headers to perform TCP Segmentation Offload type functionality on the Geneve segments. In addition, guest VMs need to enable TSO, the default behavior with most modern operating systems.

TCP Segmentation Offload (TSO): TCP Segmentation offload is a well-established TCP optimization scheme of relatively long duration that allows large segments to pass through the TCP stack, instead of smaller packets as enforced by the MTU on the physical fabric.

1.1 (TSO) Applicability to Geneve Overlay Traffic

In the context of Geneve, NICs are aware of the Geneve headers and perform TSO taking Geneve headers into consideration. The following image (Figure 3 Geneve Offload – TSO) shows how the VM would transmit 64K segments, which go through the NSX-T Components such as switching, routing and firewall and the ESX TCP Stack, as 64K segments. NIC cards take care of chopping the segments down to MTU-sized packets before moving them on to the physical fabric.

TSO's primary benefit is in reducing CPU cycles. This benefit could help provide more cycles for actual workloads and also consequently marginally drive network performance upwards.

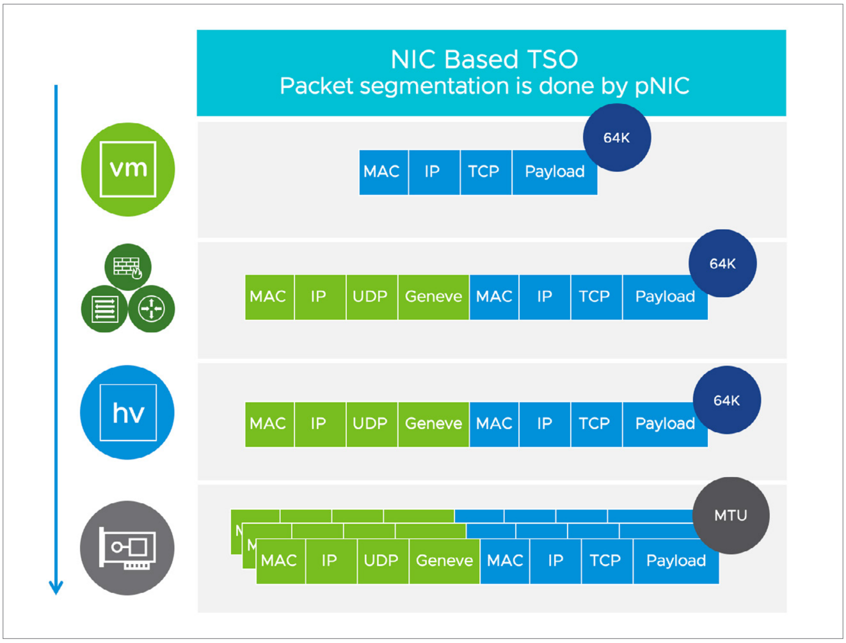

NIC Based TSO
Packet segmentation is done by pNIC
vm
64K
MAC IP TCP Payload
64K
MAC IP UDP Geneve MAC IP TCP Payload
hv
64K
MAC IP UDP Geneve MAC IP TCP Payload
MTU
MAC IP UDP Geneve MAC IP TCP Payload

Figure 3 NIC Based Geneve Offload – TSO

1.2 NIC Supportability with TSO for Geneve Overlay Traffic

In cases where the NIC card does not support TSO for Geneve overlay traffic, TSO is done in software by the hypervisor just before moving the MTU-sized packets to the NIC card. Thus, NSX-T components are still able to leverage TSO.

The following image (Figure 4 Software/CPU based Geneve Offload – TSO) shows the process where the hypervisor divides the larger TSO segments to MTU-sized packets.

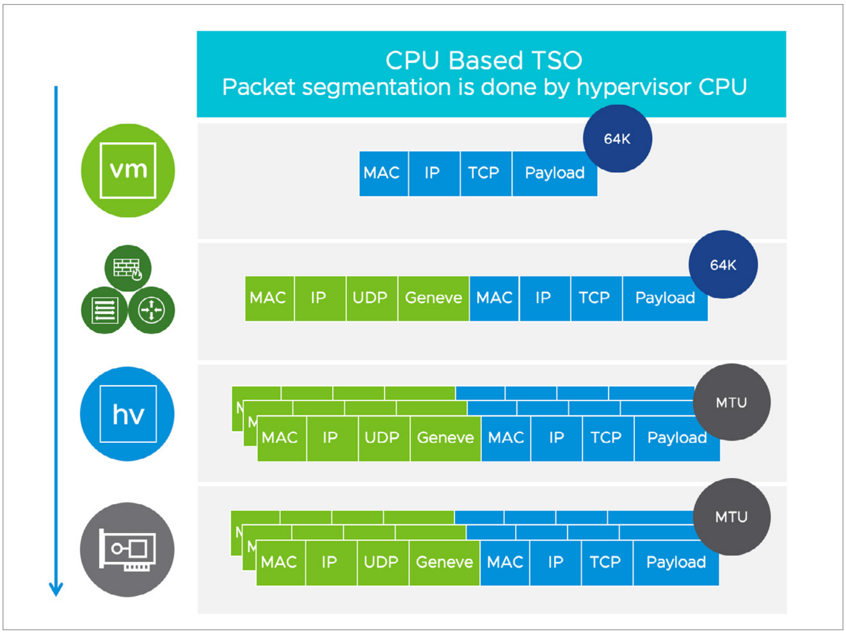


Figure 4 Software/CPU Based Geneve Offload – TSO

1.3 NIC Card Geneve Offload Capability

1.3.1 VMware's IO Compatibility Guide

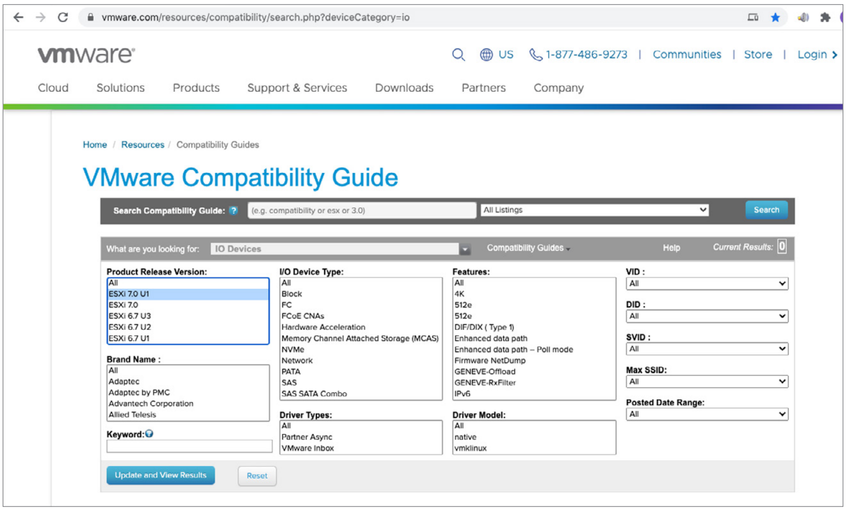
VMware's IO compatibility guide is a publicly accessible online tool:

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

VMware's IO compatibility guide is the single source of truth to confirm whether a particular card is Geneve-offload capable. It is important to note availability of Geneve offload capability in the NIC helps decrease CPU cycles and increase throughput. Hence, deploying NICs with Geneve compatibility does have marginal performance implications. However, in cases where the NIC does not have Geneve capability, ESX automatically falls back to software based Geneve offload mode. While NIC-based offload is ideal, software-based offload still helps reduce the CPU cycles spent for NSX components.

The following section show the steps to check whether a card, Intel 810s in this case, supports Geneve offload.

1. Access the online tool: https://www.vmware.com/resources/compatibility/search.php?deviceCategory=io



VMware Compatibility Guide

Search Compatibility Guide: (e.g. compatibility or esx or 3.0) All Listings

What are you looking for: IO Devices

Current Results: 0

Product Release Version:
All
ESXi 7.0 U1
ESXi 7.0
ESXi 6.7 U3
ESXi 6.7 U2
ESXi 6.7 U1

I/O Device Type:
All
Block
FC
FCoE CNAs
Hardware Acceleration
Memory Channel Attached Storage (MCAS)
NVMe
Network
PATA
SAS
SAS SATA Combo

Features:
All
4K
512e
512e
DIF/DIX (Type 1)
Enhanced data path
Enhanced data path – Poll mode
Firmware NetDump
GENEVE-Offload
GENEVE-RxFilter
IPv6

VID : All

DID : All

SVID : All

Max SSID: All

Posted Date Range: All

Brand Name :
All
Adaptec
Adaptec by PMC
Advantech Corporation
Allied Telesis

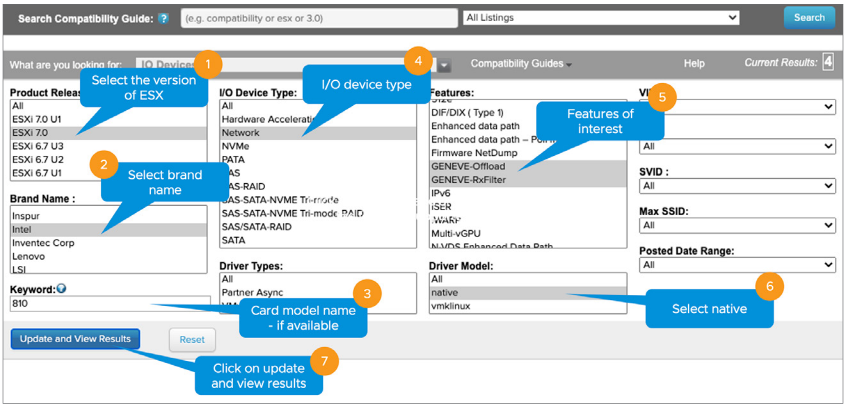
Keyword:

Driver Types:
All
Partner Async
VMware Inbox

Driver Model:
All
native
vmklinux

Figure 5 VMware Compatibility Guide for I/O

2. Specify the
    1. Version of ESX
    2. Vendor of the NIC card
    3. Model if available
    4. Select Network as the IO Device Type
    5. Select Geneve Offload and Geneve Rx Filters (more on that in the upcoming section) in the Features box
    6. Select Native
    7. Click "Update and View Results"


Search Compatibility Guide:
(e.g. compatibility or esx or 3.0)
All Listings
Search
What are you looking for: IO Devices
Compatibility Guides
Help
Current Results: 4
Select the version of ESX
1
I/O device type
4
Features of interest
5
Select brand name
2
Card model name - if available
3
Select native
6
Click on update and view results
7
Update and View Results
Reset

Product Release: All, ESXi 7.0 U1, **ESXi 7.0**, ESXi 6.7 U3, ESXi 6.7 U2, ESXi 6.7 U1

Brand Name: Inspur, **Intel**, Inventec Corp, Lenovo, LSI

Keyword: 810

I/O Device Type: All, Hardware Accelerati, **Network**, NVMe, PATA, AS, AS-RAID, SAS-SATA-NVME Tri-mode, SAS-SATA-NVME Tri-mode RAID, SAS/SATA-RAID, SATA

Driver Types: All, Partner Async

Features: DIF/DIX (Type 1), Enhanced data path, Enhanced data path – Poll, Firmware NetDump, **GENEVE-Offload**, **GENEVE-RxFilter**, IPv6, iSER, iWARP, Multi-vGPU, NVDS Enhanced Data Path

Driver Model: All, **native**, vmklinux

VID: All

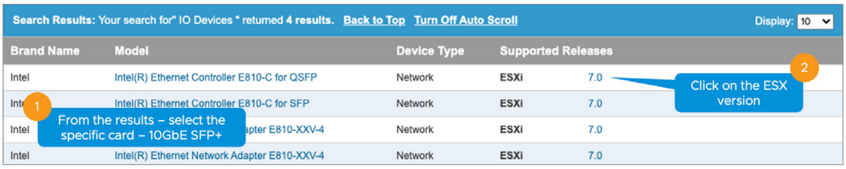
SVID: All

Max SSID: All

Posted Date Range: All

Figure 6 Steps to Verify Compatibility – Part 1

3. From the results, click on the ESX version for the concerned card. In this example, ESXi version 7.0 for Intel E810-C with QSFP ports:

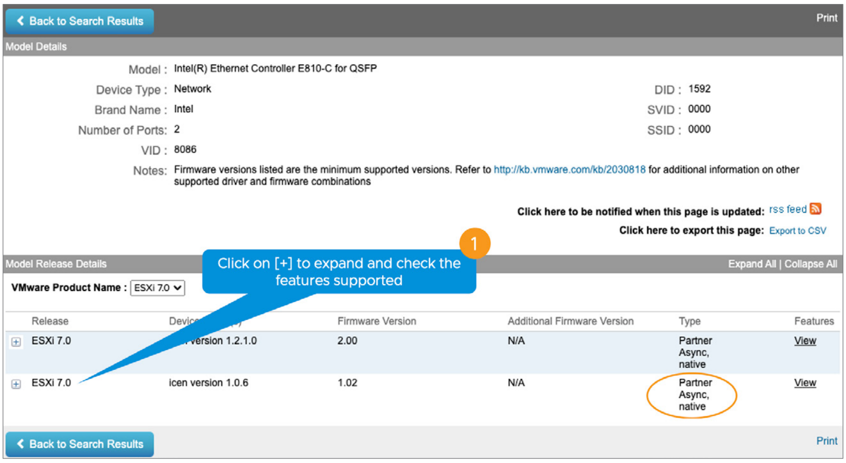


Search Results: Your search for" IO Devices " returned 4 results.

Brand Name	Model	Device Type	Supported Releases	
Intel	Intel(R) Ethernet Controller E810-C for QSFP	Network	ESXi	7.0
Int	Intel(R) Ethernet Controller E810-C for SFP	Network	ESXi	7.0
Intel	apter E810-XXV-4	Network	ESXi	7.0
Intel	Intel(R) Ethernet Network Adapter E810-XXV-4	Network	ESXi	7.0

Figure 7 Steps to Verify Compatibility – Part 2

4. Click on the [+] symbol to expand and check the features supported.


Back to Search Results
Print
Click on [+] to expand and check the features supported
1
Expand All | Collapse All
Back to Search Results
Print

Model Details

Model :	Intel(R) Ethernet Controller E810-C for QSFP
Device Type :	Network
Brand Name :	Intel
Number of Ports:	2
VID :	8086
DID :	1592
SVID :	0000
SSID :	0000
Notes:	Firmware versions listed are the minimum supported versions. Refer to http://kb.vmware.com/kb/2030818 for additional information on other supported driver and firmware combinations

Click here to be notified when this page is updated: rss feed

Click here to export this page: Export to CSV

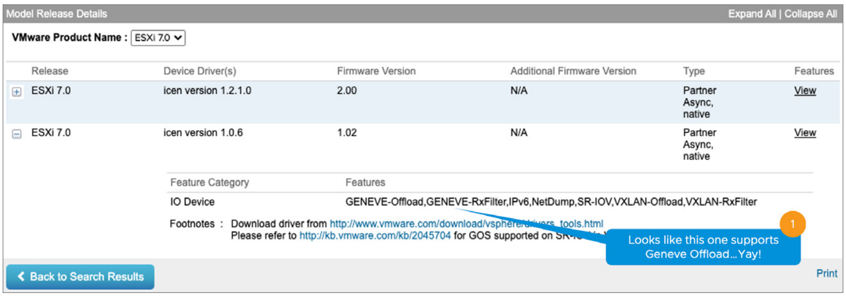
Model Release Details

VMware Product Name : ESXi 7.0

	Release	Devic	Firmware Version	Additional Firmware Version	Type	Features
⊞	ESXi 7.0	version 1.2.1.0	2.00	N/A	Partner Async, native	View
⊞	ESXi 7.0	icen version 1.0.6	1.02	N/A	Partner Async, native	View

Figure 8 Steps to Verify Compatibility – Part 3

5. Make sure the concerned driver actually has the "Geneve-Offload and Geneve-Rx Filters" as listed features.


Expand All | Collapse All
Looks like this one supports Geneve Offload...Yay!
1
Back to Search Results
Print

Model Release Details

VMware Product Name : ESXi 7.0

	Release	Device Driver(s)	Firmware Version	Additional Firmware Version	Type	Features
⊞	ESXi 7.0	icen version 1.2.1.0	2.00	N/A	Partner Async, native	View
⊟	ESXi 7.0	icen version 1.0.6	1.02	N/A	Partner Async, native	View

Feature Category	Features
IO Device	GENEVE-Offload,GENEVE-RxFilter,IPv6,NetDump,SR-IOV,VXLAN-Offload,VXLAN-RxFilter

Footnotes : Download driver from http://www.vmware.com/download/vsphere/drivers_tools.html
Please refer to http://kb.vmware.com/kb/2045704 for GOS supported on SR-

Figure 9 Steps to Verify Compatibility – Part 4

Follow the above procedure to ensure Geneve Offload and Geneve Rx Filters are available on any NIC card you are planning to deploy for use with NSX-T. As mentioned earlier, not having Geneve Offload will impact performance with higher CPU cycles spent to make up for the lack of hardware based Geneve Offload capabilities.

1.3.2 ESXi-Based Hypervisor

On an ESXi host with a NIC card supporting Geneve-Offload in Hardware with the appropriate supported driver, the following commands can be used to confirm Geneve-Offload is enabled on a pNIC — in this case pNIC vmnic3:

```
[Host-1] vsish -e get /net/pNics/vmnic3/properties | grep
".*Activated.*Geneve"

    Device Hardware Cap Activated:: 0x793c032b ->
VMNET_CAP_SG VMNET_CAP_IP4_CSUM VMNET_CAP_HIGH_DMA
VMNET_CAP_TSO VMNET_CAP_HW_TX_VLAN
VMNET_CAP_HW_RX_VLAN VMNET_CAP_SG_SPAN_PAGES
VMNET_CAP_IP6_CSUM VMNET_CAP_TSO6 VMNET_CAP_TSO256k
VMNET_CAP_ENCAP VMNET_CAP_Geneve_OFFLOAD
VMNET_CAP_IP6_CSUM_EXT_HDRS VMNET_CAP_TSO6_EXT_HDRS
VMNET_CAP_SCHED
```

CLI 1 Check Geneve Offload Support

Look for the tag "VMNET_CAP_Geneve_OFFLOAD", highlighted in red above. This verbiage indicates the Geneve Offload is activated on NIC card vmnic3. If the tag is missing, then it means Geneve Offload is not enabled because either the NIC or its driver does not support it.

Receive Side Scaling (RSS) and Rx Filters

Readers familiar with software based VxLAN deployment with NSX-V, are likely familiar with the immense performance benefits of RSS, including improving the performance of overlay traffic by four (4) times.

1.4 Benefits with RSS

RSS, another long-standing TCP enhancement, enables use of multiple cores on the receive side to process incoming traffic. Without RSS, ESX by default will use only one core to process incoming traffic. Utilizing only one core has a huge impact on the overall throughput as the receiving node then becomes the bottleneck. RSS on the NIC creates multiple queues to process incoming traffic and efficiently uses a core for each queue, with most NIC cards being able to support at least 4 queues. Hence the 4x benefit of using RSS. See Figure 10 on RSS for a visual representation of how this works.

1.4.1 RSS for Overlay

While RSS itself in general is in fairly common use today, there are NICs which still may not support RSS for overlay. Hence, our recommendation is to confirm with the NIC vendor whether RSS for overlay traffic is available in hardware, then also confirm with the VMware Compatibility IO Guide (https://www.vmware.com/resources/compatibility/search.php?deviceCategory=io) whether there is a RSS-certified driver.

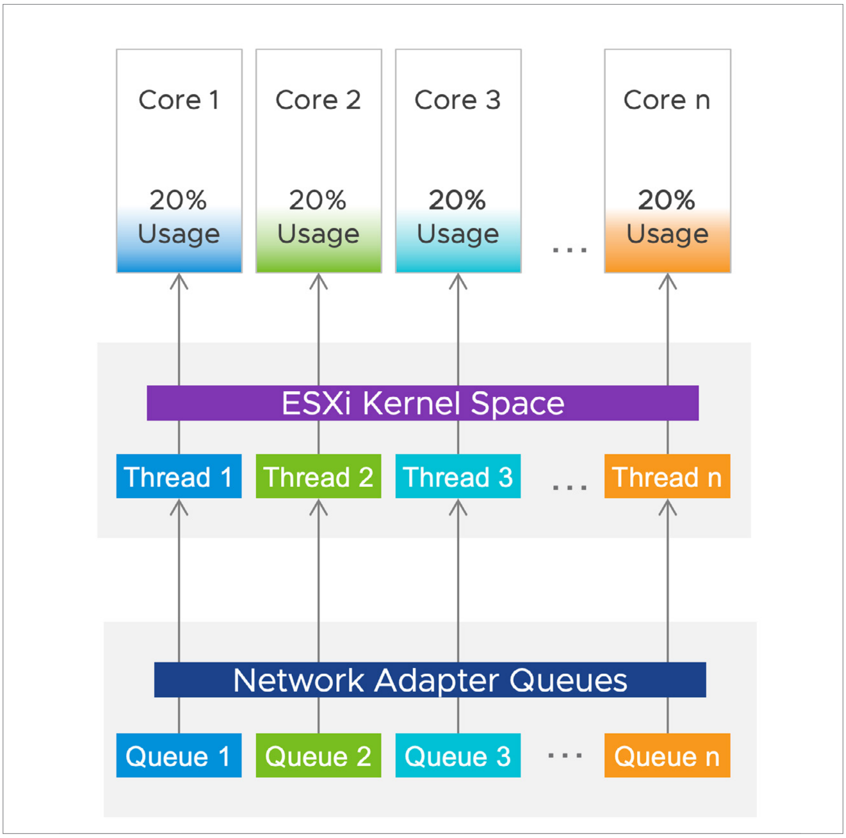


Figure 10 RSS

1.4.2 Enabling RSS for Overlay

Every vendor has their own unique mechanism to enable RSS for overlay traffic. There are also cases where the setting used to change RSS is different based on the driver version. Please refer to the concerned vendor documentation for details on enabling RSS for specific NICs.

1.4.3 Checking Whether RSS is Enabled

Use the “vsish” command to check whether RSS is enabled. The following example shows how to check whether RSS (marked blue) is enabled on NIC vmnic (marked in red).

```
[Host-1] # vsish
/> get /net/pNics/vmnic0/rxqueues/info
rx queues info {
# queues supported:5
# filters supported:126
# active filters:0
Rx Queue features:features: 0x1a0 -> Dynamic RSS Dynamic
Preemptible
}
/>
```

CLI 2 Check RSS

1.4.4 RSS and Rx Filters - Comparison

RSS uses the outer headers to hash flows to different queues. Using outer headers of Geneve overlay, especially between two hypervisors, may not be optimal as the only varying parameter is the source port.

The following image shows the fields used by RSS (circled in red) to hash flows across various CPU cores. Since all the fields are from the outer headers, there is a little variability and in the worst-case scenarios the only variable may be the source port.

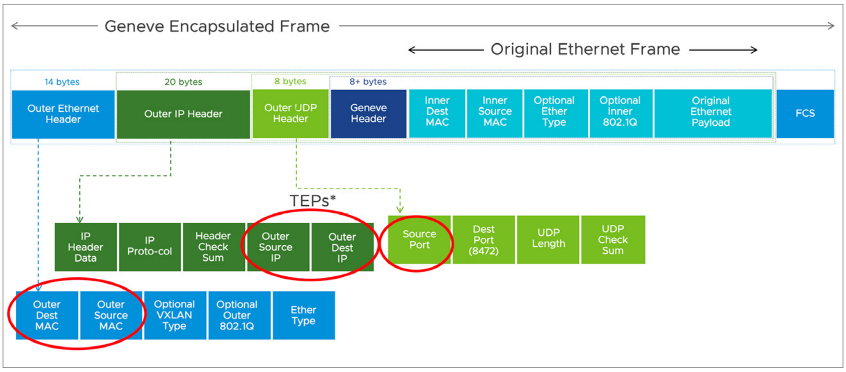


Figure 11 RSS: Fields Used for Hashing

To overcome this limitation, the latest NICs (see compatibility guide) support an advanced feature, known as Rx Filters, which looks at the inner packet headers for hashing flows to different queues on the receive side. In the following image (Figure 12), fields used by Rx Filter are circled in red.

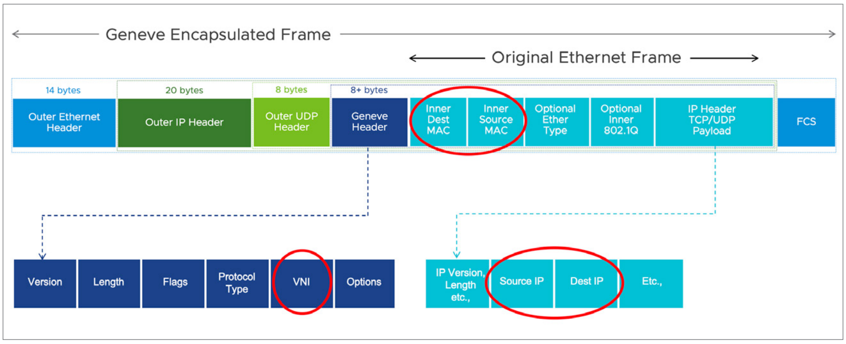


Figure 12 Rx Filters: Fields Used for Hashing

Simply put, Rx Filters look at the inner packet headers for queuing decisions. The queuing decision itself is based on flows, bandwidth utilization and is driven by NSX. Hence, Rx Filters provide optimal queuing compared to RSS, which is akin to a hardware-based brute force method.

1.4.5 Checking whether Rx Filters are enabled

Rx Filters are enabled by default on a NIC that supports Rx Filters in hardware and has a driver to use it. Please use the VMware's Compatibility Guide for IO, discussed earlier, to confirm whether Rx Filters are available. In VCG for I/O page, select "Geneve-RxFilter", and make sure the right driver is installed on the ESXi host.

On the ESXi host, use the "vsish" command to check whether Rx Filters are enabled. The following example shows how to check whether the NIC vmnic5 (marked in red) has Rx Filters Enabled for Geneve (marked in blue)

Check Whether Rx / Tx Filters are Enabled:

```
[Host-1] vsish
/> cat /net/pNics/vmnic5/rxqueues/info
rx queues info {
   # queues supported:8
   # filters supported:512
   # active filters:0
   # filters moved by load balancer:254
   # of Geneve OAM filters:2
```

```
    RX filter classes:Rx filter class: 0x1c -> VLAN_MAC
VXLAN Geneve GenericEncap
    Rx Queue features:features: 0x82 -> Pair Dynamic
}
/>
```

CLI 3 Check RxFilters Support

1.4.6 RSS vs Rx Filters for Edge VM

In the case of Edge VMs, the hypervisor does not encapsulate/decapsulate the overlay packets. Instead, the packets are sent along with the overlay headers to the Edge VM's vNIC interfaces. In this case, RSS is the best mechanism available today to hash packets to separate queues. See more on RSS for Edges in the Edge sections.

Jumbo MTU for Higher Throughput

Maximum Transmission Unit (MTU) denotes the maximum packet size that can be sent on the physical fabric. When setting this configuration on the ESX hosts and the physical fabric, Geneve header size has to be taken into consideration. Our general recommendation is to allow for at least 200 bytes buffer for Geneve headers in order to accommodate the option field for use cases such as service-insertion. As an example, if the VM's MTU is set to 1500 bytes, pNIC and the physical fabric should be set to 1700 or more. See Figure 13 on MTU Configuration.

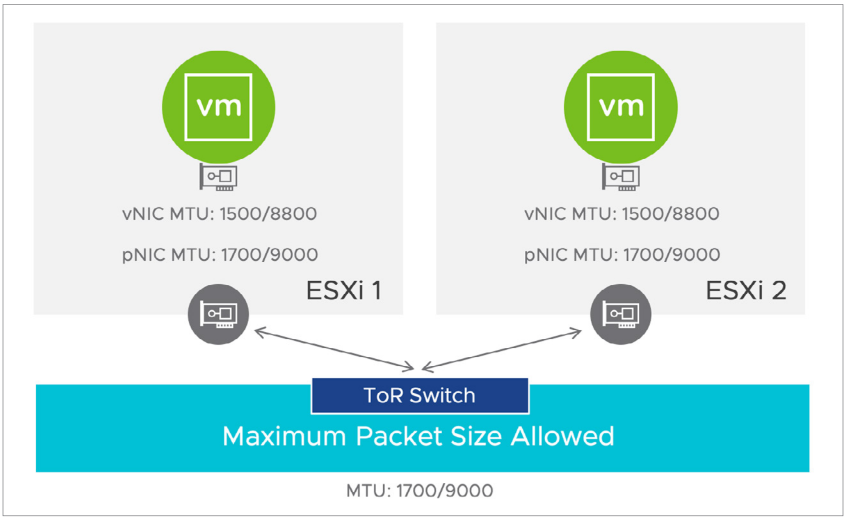


Figure 13 MTU Configuration

Why should you care about MTU values? MTU is a key factor for driving high throughput, and this is true for any NIC which supports 9K MTU also known as jumbo frame. The following graph (Figure 14 MTU and Throughput) shows throughput achieved with MTU set to 1500 and 8800:

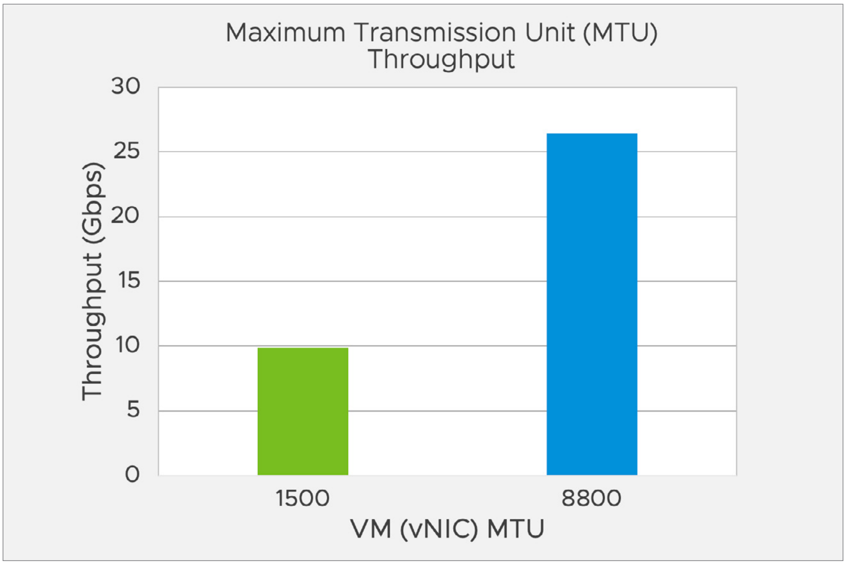


Figure 14 MTU and Throughput

Our recommendation for optimal throughput is to set the underlying fabric and ESX host's pNICs to 9000 and the VM vNIC MTU to 8800.

Notes for Figure 14 MTU and Throughput:

- The above graph represents a single pair of VMs running iPerf with 4 sessions.
- For both VM MTU cases of 1500 and 8800, the MTU on the host was 9000 with demonstrated performance improvements.

1.5 Checking MTU on an ESXi Host

Use the *esxcfg-nics* command to check the MTU:

```
[Host-1] esxcfg-nics -l | grep vmnic5
vmnic5 0000:84:00.0 i40en     Up    40000Mbps  Full
 3c:fd:fe:9d:2b:d0 9000    Intel Corporation Ethernet Controller XL710
for 40GbE QSFP+
```

CLI 4 Check MTU on ESXi Host

For the VM, use the commands specific to the operating system for checking MTU. For example, “ifconfig” is one of the commands in Linux to check MTU.

Single TEP vs Dual TEP

Dual TEP is another method to increase the throughput of a given host. Dual TEP will help achieve twice the throughput of single TEP. The following image compares throughput with Single TEP vs throughput with Dual TEP using Intel® XL710 NICs on servers running on Intel® Xeon® Gold 6252 CPU @ 2.10GHz.

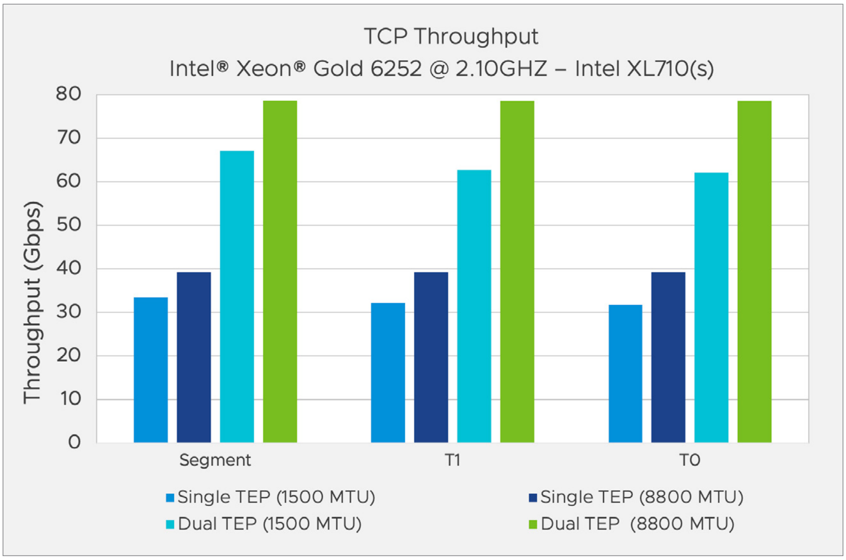


Figure 15 Throughput with Single TEP vs Dual TEP with Different MTUs

Note: Intel® XL710s are PCIe Gen 3 x8 lane NICs. On x8 lane NICs, the max throughput is limited to 64Gbps. To achieve the above near-line rate of 80Gbps, 2 x Intel® XL710s must be used.

NIC Port Speed

As a final consideration, the port speed of the NIC, in combination with all the above features, also has a direct impact on the achieved throughput. The following graph shows throughput achieved on an Intel® E810C

100Gbps NIC. This is a PCIe Gen 4 x16 lane NIC running on a PCIe Gen 3 platform. Thanks to its x16 lane design, this NIC's performance is close to line rate at 100Gbps with MTU set to 8800 on the VMs.

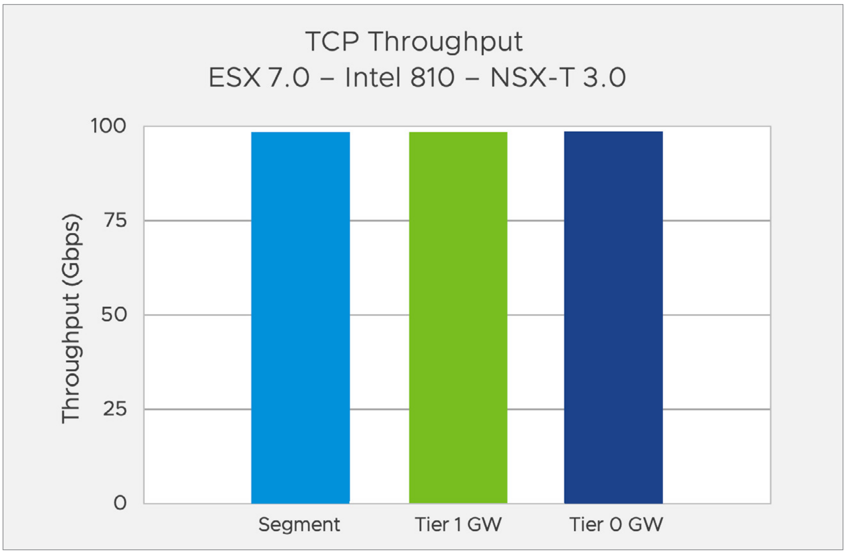


Figure 16 Throughput with Intel® E810-C 100Gbps NIC

Performance Factors for NSX-T Edges

1.6 Data Plane Development Kit (DPDK)

Data Plane Development Kit (DPDK) is a set of libraries and drivers to enable fast packet processing on a wide variety of CPU architectures including x86 platforms. DPDK is applicable for both the VM and bare metal Edge form factors, with the VM edge capable of delivering over 20Gbps throughput for standard TCP based DC workloads. Bare metal Edge delivers over 35Gbps throughput for the same TCP-based DC workloads. The bare metal Edge form factor also excels at processing small packet sizes ~78 Bytes at near line on a 40Gbps port such as Intel® XL710, which is useful in NFV-style workloads. The following graph (Figure 17 Bare Metal Edge Performance Test (RFC2544) with IXIA) shows the performance of bare metal Edge with a standard RFC 2544 test with IXIA.

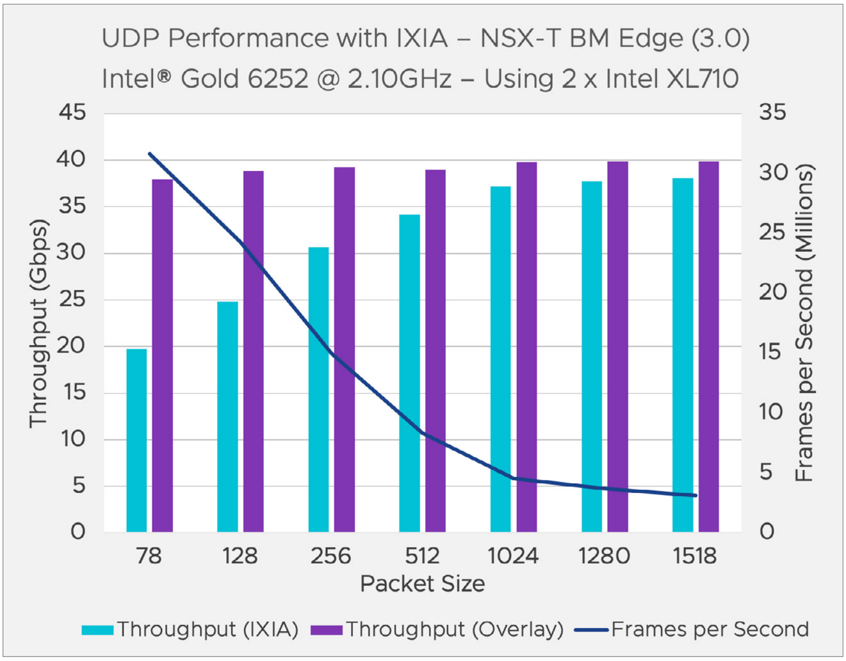


Figure 17 Bare Metal Edge Performance Test (RFC2544) with IXIA

Note: For the above test, the overlay lines in purple are calculated by adding throughput reported by IXIA and the Geneve Overlay header size.

1.7 SSL Offload

VMware NSX-T bare metal Edges also support SSL Offload. This configuration helps in reducing the CPU cycles spent on SSL Offload and also has a direct impact on the throughput achieved. In the following image, Intel® QAT 8960s are used to show case the throughput achieved with SSL Offload.

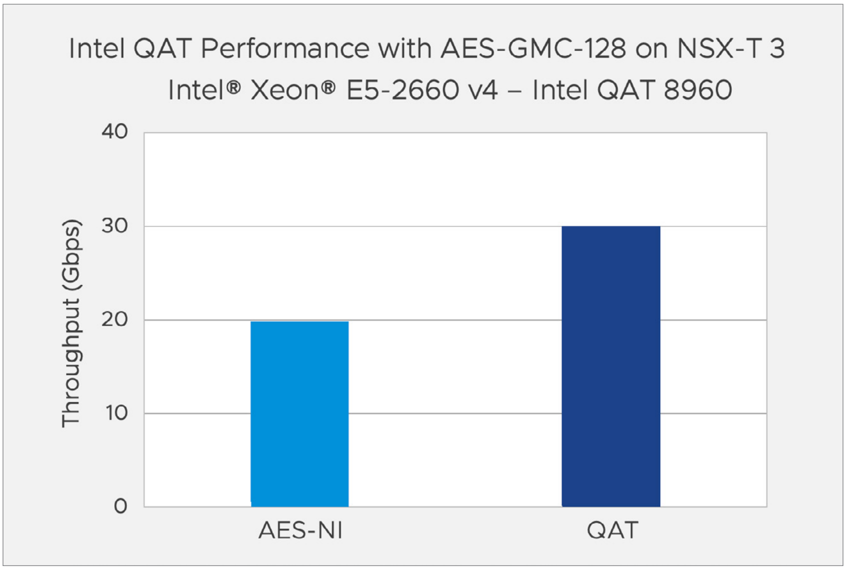


Figure 18 Throughput with SSL Offload

Key Performance Factors

One key to achieve better performance for compute, Edge VM, and bare metal Edge is to ensure having the right set of tools. While DPDK plays a key role, it's not the only factor affecting performance. The combination of DPDK and other capabilities of NIC drivers and hypervisor enhancements will help achieve optimal performance.

1.8 Compute Node Performance Factors

For compute clusters, our recommendation is to ensure the following two features are available on your NIC of choice:

1. Geneve Offload
2. Geneve Rx Filters

Geneve Offload helps decrease the CPU usage by offloading the task of dividing TSO segments into MTU-determined packets to the NIC card while also helping to increase throughput. Geneve Rx Filters help increase the number of cores used to process incoming traffic, which is in turn increases performance by a factor of 4x times based on the number of hardware queues available on the NIC. For example, on Intel XL710s, throughput is improved to near line rate, thanks to Geneve Rx Filters feature. For older cards which do not support Geneve Rx Filters, check whether they at minimum have RSS capability. The following graph shows the throughput achieved with and without using Geneve Rx Filters, 10Gbps vs near line rate:

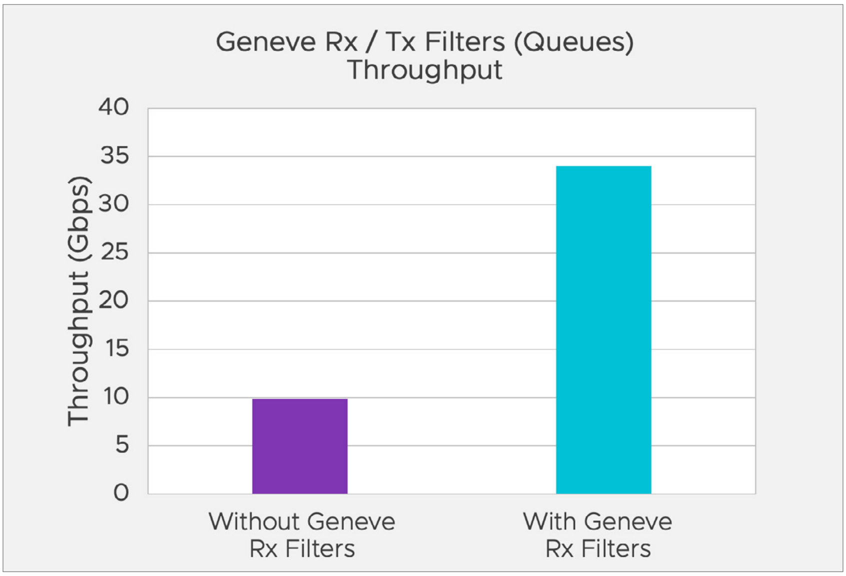


Figure 19 Throughput Improvement with Rx Filters

Figure 19 note: This test was run with Large Receive Overload (LRO) enabled, which is software-supported starting with ESX version 6.5 and higher on the latest NICs which support the Geneve Rx Filter. Thus, along with Rx Filters, LRO contributes to the higher throughput depicted here.

1.9 VM Edge Node Performance Factors

In the case of edge clusters, DPDK is the key factor for performance. In the case of VM Edge, RSS-enabled NICs are best for optimal throughput.

RSS at pNIC

To achieve the best throughput performance, use an RSS-enabled NIC on the host running Edge VM, and ensure an appropriate driver which supports RSS is also being used. Use the VMware Compatibility Guide for I/O (section 1.3.1.) to confirm driver support.

RSS on VM Edge

For best results, enable RSS for Edge VMs. Following is the process to enable RSS on Edge VMs:

1. Shutdown the Edge VM
2. Find the ".vmx" associated with the Edge VM (https://kb.vmware.com/s/article/1714)
3. Change the following two parameters, for the Ethernet devices in use, inside the .vmx file
    a. ethernet3.ctxPerDev = "3"
    b. ethernet3.pnicFeatures = "4"
4. Start the Edge VM

Alternatively, use the vSphere Client to make the changes:

1. Right click on the appropriate Edge VM and click "Edit Settings":

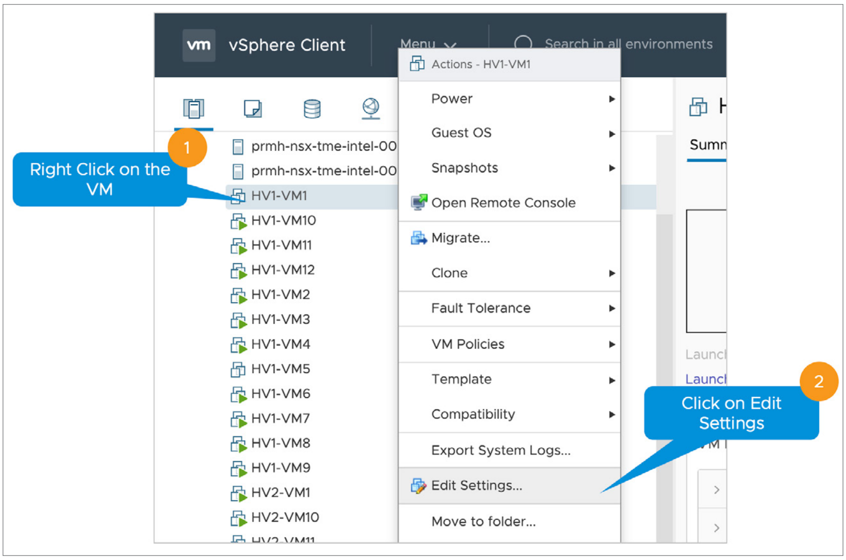
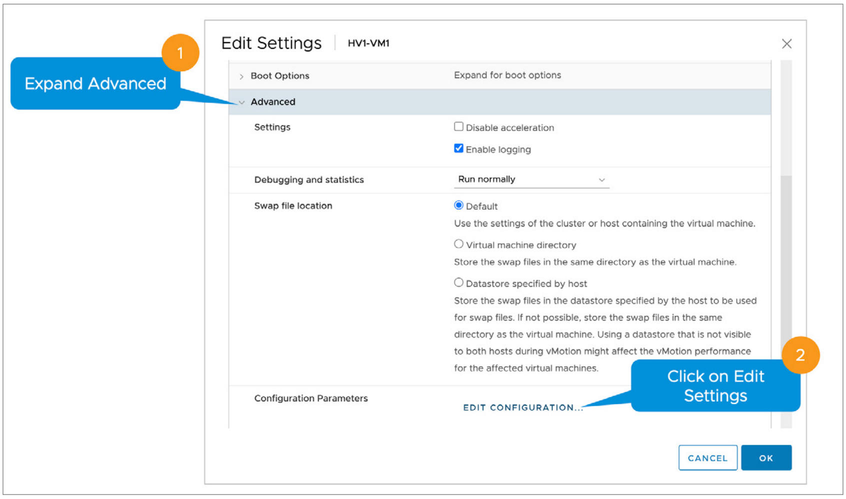


Figure 20 Change VM RSS Settings via vSphere Client – Part 1

2. Under “VM Options”, expand “Advanced” and click on “Edit Configuration”:



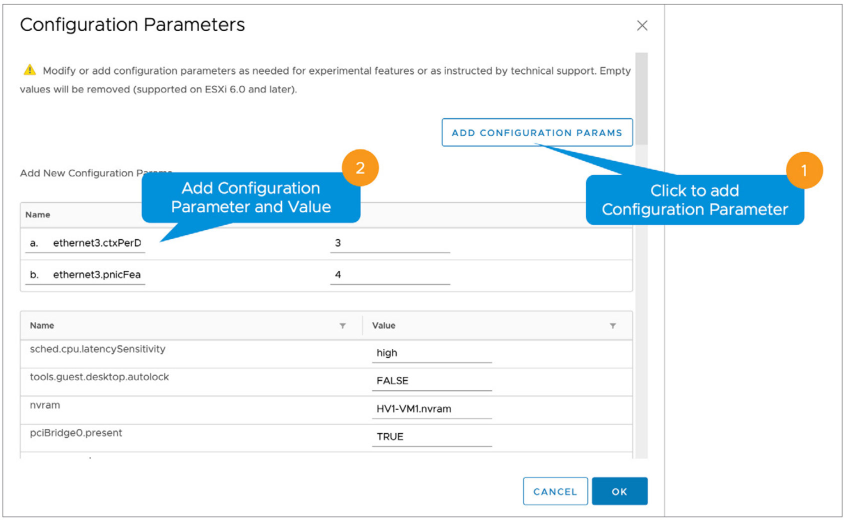
Edit Settings | HV1-VM1

› Boot Options	Expand for boot options
⌄ Advanced	
Settings	☐ Disable acceleration ☑ Enable logging
Debugging and statistics	Run normally
Swap file location	● Default Use the settings of the cluster or host containing the virtual machine. ○ Virtual machine directory Store the swap files in the same directory as the virtual machine. ○ Datastore specified by host Store the swap files in the datastore specified by the host to be used for swap files. If not possible, store the swap files in the same directory as the virtual machine. Using a datastore that is not visible to both hosts during vMotion might affect the vMotion performance for the affected virtual machines.
Configuration Parameters	EDIT CONFIGURATION...

CANCEL OK

Figure 21 Change VM RSS Settings via vSphere Client – Part 2

3. Add the two configuration parameters by clicking on “Add Configuration” for each item to add:



2
Add Configuration Parameter and Value
1
Click to add Configuration Parameter

Configuration Parameters

⚠ Modify or add configuration parameters as needed for experimental features or as instructed by technical support. Empty values will be removed (supported on ESXi 6.0 and later).

ADD CONFIGURATION PARAMS

Add New Configuration Params

Name	Value
a. ethernet3.ctxPerD	3
b. ethernet3.pnicFea	4

Name	Value
sched.cpu.latencySensitivity	high
tools.guest.desktop.autolock	FALSE
nvram	HV1-VM1.nvram
pciBridge0.present	TRUE

CANCEL OK

Figure 22 Change VM RSS Settings via vSphere Client – Part 3

The following graph (Figure 23 RSS for VM Edge) shows the comparison of throughput between a NIC which supports RSS and a NIC which does not. **Note:** In both cases, even where the pNIC doesn't support RSS, RSS was enabled on the VM Edge:

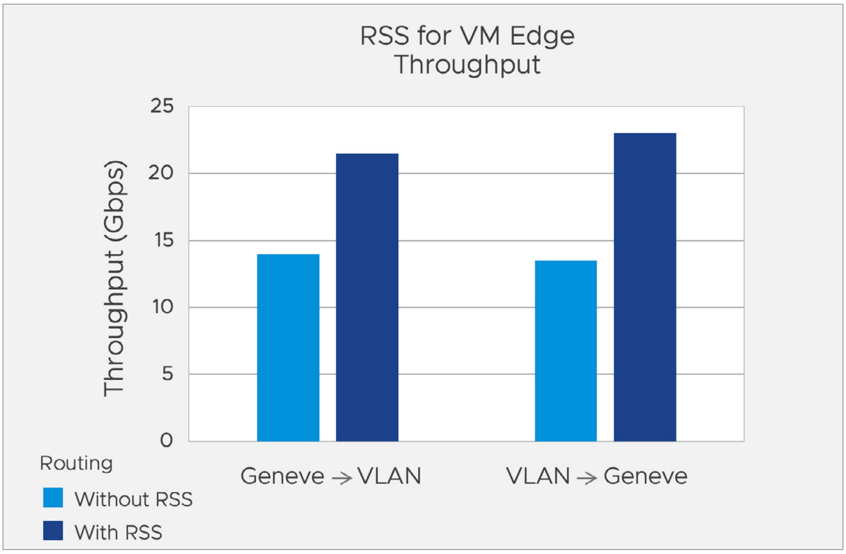


Figure 23 RSS for VM Edge

With an RSS-enabled NIC, a single Edge VM may be tuned to drive over 20Gbps throughput. As the above graph shows, RSS may not be required for 10Gbps NICs as they can achieve close to ~15 Gbps throughput even without enabling RSS.

1.10 Bare Metal Edge Node Performance Factors

Bare metal Edge has specific requirements dependent on the type of NIC used. Please refer to the NSX-T Installation Guide (https://docs.vmware.com/en/VMware-NSX-T-Data-Center/3.0/installation/GUID-14183A62-8E8D-43CC-92E0-E8D72E198D5A.html) for details on currently supported cards.

While VM and bare metal Edges leverage Data Path Development Kit (DPDK), they differ in deployment. While VM Edge does not have restrictions on the physical NIC which can be used because VMXNET3 provides DPDK functionality for the VM Edge, bare metal Edge does have strict restrictions on the NICs which may be used. (Note however, bare metal Edge nodes do support Intel® QATs for SSL Offload.) Please refer to the relevant hardware requirements section of the NSX-T installation guide for specific details on compatibility.

The following link shows the general system requirements for NSX-T 3.0 components: https://docs.vmware.com/en/VMware-NSX-T-Data-Center/3.0/installation/GUID-14183A62-8E8D-43CC-92E0-E8D72E198D5A.html

From the above page, for requirements specific to the NSX-T bare metal Edge, click on NSX Edge Bare Metal Requirements.

1.10.1 Summary of Performance – NSX-T Components to NIC Features

The following table (Table 1 Summary of Performance Impact with Various Capabilities) provides an overall view and guidance on NSX-T components and NIC features per given use case. For common datacenter applications and deployments, Standard N-VDS is the recommendation. Enhanced Data Path is only meant for NFV style workloads with fast packet processing requirements.

	Compute Transport Nodes (N-VDS Standard)	**Compute Transport Nodes (“Enhanced Data Path”)**	**ESXi nodes with VM Edges**	**Bare Metal Edge**
Features that Matter	**Geneve-Offload:** To save on CPU cycles **Geneve-RxFilters:** To increase throughput by using more cores and using software based LRO **RSS** (if Geneve-RxFilters does not exist): To increase throughput by using more cores	**N-VDS Enhanced Data Path:** For DPDK-like capabilities	**RSS:** To leverage multiple cores	**DPDK:** Poll mode driver with memory- related enhancements to help maximize packet processing speed **QATs:** For high encrypt/decrypt performance with SSL-offload
Benefits	High Throughput for typical TCP-based DC Workloads	Maximum PPS for NFV style workloads	Maximize Throughput for typical TCP based Workloads with Edge VM VM Tuning + NIC with RSS Support Add/Edit two parameters to the Edge VM’s vmx file and restart	Maximum PPS Maximum Throughput even for small packet sizes Maximum encrypt/decrypt performance with SSL Offload Low latency Maximum Scale Fast Failover

Table 1 Summary of Performance Improvements with Various Capabilities

1.11 Results

The following section takes a look at the results achievable under various scenarios with hardware designed for performance. First, here are the test bed specs and methodology:

Compute	Virtual Machine	Edge Bare Metal	Test Tools
• CPU: Intel(R) Xeon(R) Gold 6252 CPU @ 2.10GHz • RAM: 192 GB • Hyper Threading: Enabled • MTU: 1700 • NIC: XL710 • NIC Driver: i40e - 1.3.1-18vmw.670.0.0.8169922 • ESXi 6.7	vCPU: 2 RAM: 2 GB Network: VMXNET3 MTU: 1500	CPU: Intel ® Xeon ® E5-2637 v4 3.5Ghz • RAM: 256 GB • Hyper Threading: Enabled • MTU: 1700 • NIC: XL710 • NIC Driver: In-Box	iPerf 2 (2.0.5) with • 4 – 12 VM Pairs • 4 Threads per VM Pair • 30 seconds per test • Average over three iterations

Table 2 Specific Configuration of Performance Results

NSX-T Components

- Segments
- T1 Gateways
- T0 Gateways
- Distributed Firewall: Enabled with default rules
- NAT: 12 rules – one per VM
- Bridging
    a. Six Bridge Backed Segments
    b. Six Bridge Backed Segments + Routing

The following graph shows in every scenario above, NSX-T throughput performance stays consistently close to line rate on an Intel® XL710 40Gbps NIC.

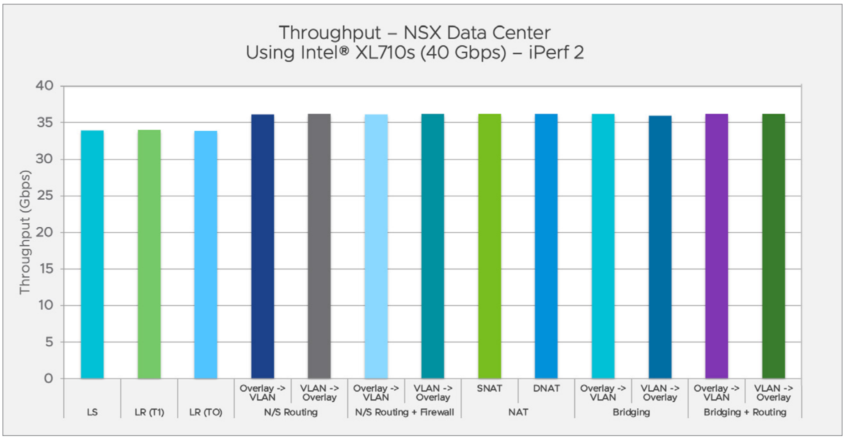


Figure 24 Throughput Summary for NSX-T Based Datacenter

Network Function Virtualization (NFV): Raw Packet Processing Performance

TCP-based workloads are generally optimized for throughput and are not sensitive to raw packet processing speed. NFV-style workloads are on the opposite end where the raw packet processing is key. For these specific workloads, NSX-T provides an enhanced version of (NSX-T Virtual Distributed Switch) called N-VDS enhanced.

1.12 N-VDS Enhanced Data Path

Based on the DPDK-like features such as Poll Mode Driver (PMD), CPU affinity, and optimization and buffer management, N-VDS Enhanced caters to applications requiring high speed raw packet processing, for details on the N-VDS enhanced switch and its application, please refer to the resource below:

Acceleration with N-VDS in Enhanced Data Path Mode

https://docs.vmware.com/en/VMware-vCloud-NFV-OpenStack-Edition/3.1/vmware-vcloud-nfv-openstack-edition-ra31/GUID-0695F86B-20AD-4BFB-94E0-5EAF94087758.html?hWord=N4IghgNiBcIHIFoBqARAygAgKIDsAWYOAxgKYAmIAvkA

1.12.1 Poll Mode Driver

One of the recent key changes with DPDK is the addition of Poll Mode Driver (PMD). With the Poll Mode Driver, instead of the NIC sending an interrupt to the CPU once a packet arrives, a core is assigned to poll the NIC to check for packets. This polling procedures eliminates CPU context switching, unavoidable in the traditional interrupt mode of packet processing, resulting in higher packet processing performance.

1.12.2 CPU Affinity and Optimization

With DPDK, dedicated cores are assigned to process packets. This assignment procedure ensures consistent latency in packet processing and enables instruction sets such SSE, which helps with floating point calculations, to be available where needed.

1.12.3 Buffer Management

Buffer management is optimized to represent the packets being processed in simpler fashion with low footprint, assisting with faster memory allocation and processing. Buffer allocation is also Non-uniform memory access (NUMA) aware. NUMA awareness reduces traffic flows between the NUMA nodes, thus improving overall throughput.

Instead of requiring regular packet handlers for packets, Enhanced Datapath uses mbuf, a library to allocate and free buffers resulting in packet-related info with low overhead. As traditional packet handlers have heavy overhead for initialization, mbuf simplifies packet descriptors by decreasing the CPU overhead for packet initialization. To further support the mbuf-based packet, VMXNET3 has also been enhanced. In addition to the above DPDK enhancements, ESX TCP Stack has also been optimized with features such as Flow Cache.

1.12.4 Flow Cache

Flow Cache is an optimization enhancement which helps reduce CPU cycles spent on known flows. With the start of a new flow, Flow Cache tables are immediately populated. This procedure enables follow-up decisions for the rest of packets within a flow to be skipped if the flow already exists in the flow table. Flow Cache uses two mechanisms to figure out fast path decisions for packets in a flow:

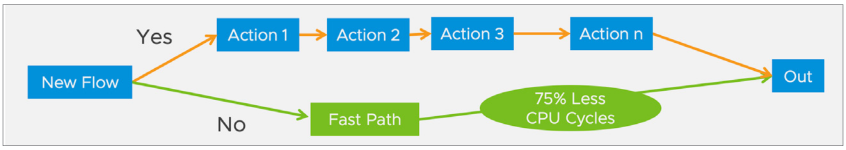


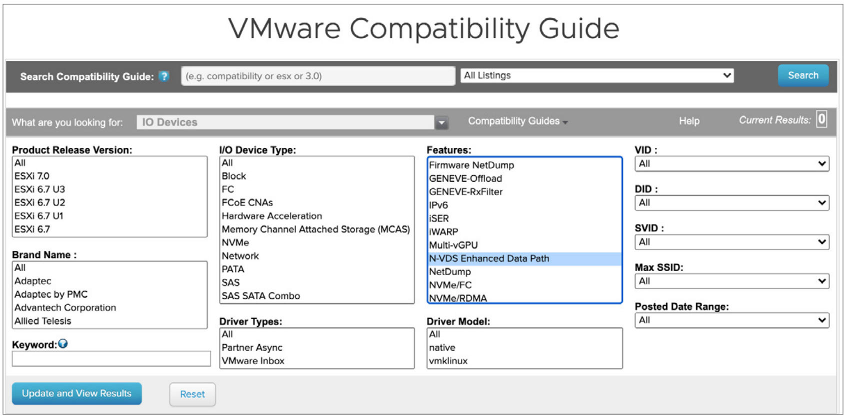
Figure 25 Flow Cache Pipeline

If the packets from the same flow arrive consecutively, the fast path decision for that packet is stored in memory and applied directly for the rest of the packets in that cluster of packets.

If packets are from different flows, the decision per flow is saved to a hash table and used to decide the next hop for packets in each of the flows. Flow Cache helps reduce CPU cycles by as much as 75%, a substantial improvement.

1.12.5 Checking whether a NIC is N-VDS Enhanced Data Path Capable

Use the VMware IO Compatibility Guide (VCG I/O) described in the previous sections to find out which cards are N-VDS (E) capable. The feature to look for is "N-VDS Enhanced Data Path" highlighted in blue in the following image:



VMware Compatibility Guide

Search Compatibility Guide: (e.g. compatibility or esx or 3.0) All Listings Search

What are you looking for: IO Devices Compatibility Guides Help Current Results: 0

Product Release Version:
All
ESXi 7.0
ESXi 6.7 U3
ESXi 6.7 U2
ESXi 6.7 U1
ESXi 6.7

Brand Name :
All
Adaptec
Adaptec by PMC
Advantech Corporation
Allied Telesis

Keyword:

I/O Device Type:
All
Block
FC
FCoE CNAs
Hardware Acceleration
Memory Channel Attached Storage (MCAS)
NVMe
Network
PATA
SAS
SAS SATA Combo

Driver Types:
All
Partner Async
VMware Inbox

Features:
Firmware NetDump
GENEVE-Offload
GENEVE-RxFilter
IPv6
iSER
iWARP
Multi-vGPU
N-VDS Enhanced Data Path
NetDump
NVMe/FC
NVMe/RDMA

Driver Model:
All
native
vmklinux

VID : All

DID : All

SVID : All

Max SSID: All

Posted Date Range: All

Update and View Results Reset

Figure 26 VMware Compatibility Guide for I/O - Selection Step for N-VDS Enhanced Data Path

Note: N-VDS Enhanced Data Path cannot share the pNIC with N-VDS - they both need a dedicated pNIC.

Benchmarking Tools

1.13 Compute

On the compute side, our recommendation for testing the software components is to use a benchmarking tool close to the application layer. Application layer benchmarking tools will help take advantage of many features and show the true performance characteristics of the system.

While application benchmarking tools are ideal, they may not be very easy to setup and run. In such cases, iPerf is a great tool to quickly setup and check throughput. Netperf is another tool to help check both throughput and latency.

Here is a github resource for an example script to run iPerf on multiple VMs simultaneously and summarize results: https://github.com/vmware-samples/nsx-performance-testing-scripts

1.14 Edges

1.14.1 VM Edge

As VM Edges are designed for typical DC workloads, application layer tools are best for testing VM Edge performance.

1.14.2 Bare Metal Edge

With the Bare Metal Edges, either application layer benchmarking tools or typical network benchmarking and packet generation tools of choice, such as Keysight PathWave (formerly IXIA) or Spirent Network Emulator may be used. One of the challenges with using a hardware benchmarking tool is creating a topology that would profile Geneve encap/decap. Following is a topology with two bare metal edges, each within its own cluster. A segment, called a crosslink segment in this example, connects them both over overlay. PathWave sends and receives only non-overlay packets. However, the segment between the two Edge clusters forces the packets to use overlay. Check the image below for details.

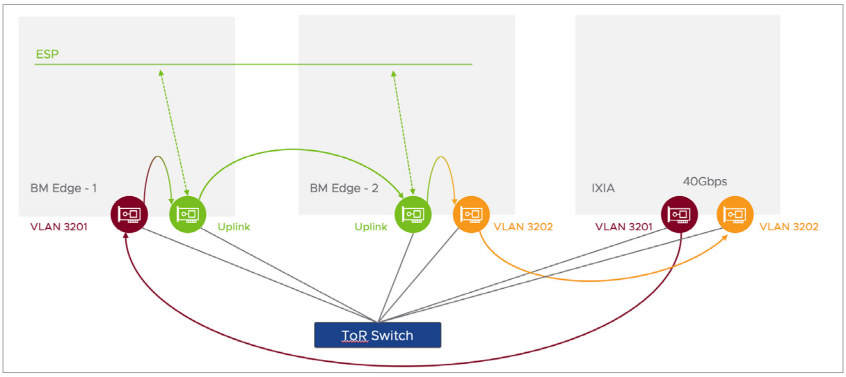


Figure 27 Example Topology to Use Geneve Overlay with Hardware IXIA or Spirent

An alternative approach is to use a software tool such as Pktgen (https://github.com/pktgen/Pktgen-DPDK).

Conclusion

To drive enhanced performance, VMware NSX-T uses a number of features supported in hardware.

On the compute side, these are:

1. Geneve Offload for CPU cycle reduction and marginal performance benefits
2. Geneve Rx Filters, an intelligent queuing mechanism to multiply throughput
3. RSS an older hardware-based queuing mechanism—alternative if Rx Filters are missing
4. Jumbo MTU an age-old trick to enable high throughput on NICs lacking above features
5. NIC port speed
6. Number of NICs—single vs dual TEP
7. PCIe Gen 3 with x16 lane NICs or multiple PCIe Gen 3 x8 NICs
8. PCIe Gen 4

For compute workloads focused on high packet processing rate for primarily small packet sizes, Enhanced Data Path Enabled NICs provide a performance boost.

For the Edges, if they are VM Edges, then:

1. NIC cards which support RSS and
2. Enabling RSS on both the pNIC and the Edge VM NIC (vNIC)

For the bare metal Edges, leveraging optimal SSL Offload performance such as Intel® QAT 8960s and deploying supported hardware from the VMware NSX-T install guide will result in performance gains.

This eBook, “VMware NSX-T® Performance Considerations on Intel® Platforms”, provides knowledge and guidance for optimizing the performance of the VMware NSX component of a software-defined data center (SDDC). To fully leverage software-defined capabilities, however, hardware-level factors influencing performance must also be considered. This eBook thus has information on both hardware-level features (such as Geneve Rx filters) as well as software-level configurations, to maximize performance.

VMware NSX — the network virtualization platform which enables the implementation of virtual networks on your physical network and within your virtual server infrastructure — has already helped over a thousand organizations improve the network and security posture of their SDDCs by fundamentally changing the approach to network and security.

“VMware NSX-T® Performance Considerations on Intel® Platforms” is your roadmap to fully realizing the benefits provided by running VMware NSX. You will find proven insights and recommendations for maximizing the performance of VMware NSX environments, unlocking their full potential to deliver previously unheard-of levels of flexibility and agility.

ISBN-13: 978-1-7358347-0-2

ISBN 978-1-7358347-0-2

9 781735 834702 >