ï»¿INTERNET-DRAFT

BMWG S. Kommu
Internet-Draft V Mware
Intended status: Informational J. Rapp
Expires: Sep 2019 V Mware

Mar 11, 2019

Considerations for Benchmarking Network Virtualization Platforms
draft-skommu-bmwg-nvp-03.txt

Status of this Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

The list of current Internet-Drafts can be accessed at http://www.ietf.org/ietf/1id-abstracts.txt

The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html

This Internet-Draft will expire on September 11, 2019.

Copyright Notice

Abstract

Current network benchmarking methodologies are focused on physical networking components and do not consider the actual application layer traffic patterns and hence do not reflect the traffic that virtual networking components work with when using network virtualization overlays (NVO3). The purpose of this document is to distinguish and highlight benchmarking considerations when testing and evaluating virtual networking components in the data center.

Table of Contents

1. Introduction

Datacenter virtualization that includes both compute and network virtualization is growing rapidly as the industry continues to look for ways to improve productivity, flexibility and at the same time cut costs. Network virtualization is comparatively new and expecte
d
to grow tremendously similar to compute virtualization. There are multiple vendors and solutions out in the market. Each vendor ofte
n
has their own recommendations on how to benchmark their solutions thus making it difficult to perform a apples-to-apples comparision between different solutions. Hence, the need for a vendor, product and cloud agnostic way to benchmark network virtualization solution
s
to help with comparison and make informed decisions when it comes t
o
selecting the right network virtualization solution.

Applications traditionally have been segmented using VLANs and ACLs between the VLANs. This model does not scale because of the 4K sca
le
limitations of VLANs. Overlays such as VXLAN were designed to address the limitations of VLANs.

With VXLAN, applications are segmented based on VXLAN encapsulation (specifically the VNI field in the VXLAN header), which is similar
to
VLAN ID in the 802.1Q VLAN tag, however without the 4K scale limitations of VLANs. For a more detailed discussion on this subje

ct

please refer RFC 7364 ’Problem Statement: Overlays for Network Virtualization’.

VXLAN is just one of several Network Virtualization Overlays (NVO). Some of the others include STT, Geneve and NVGRE. STT and Geneve have expanded on the capabilities of VXLAN. Please refer IETF’s nv

o3

working group < https://datatracker.ietf.org/wg/nvo3/documents/> for more information.

Modern application architectures, such as Micro-services, because of IP based connectivity within the app, place high demands on the networking and security when compared to the traditional three tier app models such as web, app and db. Benchmarks MUST consider whether the proposed solution is able to scale up to the demands of such applications and not just a three-tier architecture.

The benchmarks will be utilizing the various terminology and definitions from the NVO3 working group including RFC 8014 and RFC 8394.

2. Conventions used in this document

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

3. Definitions

3.1. System Under Test (SUT)

Traditional hardware based networking devices generally use the device under test (DUT) model of testing. In this model, apart from any allowed configuration, the DUT is a black box from a testing perspective. This method works for hardware based networking devices since the device itself is not influenced by any other components outside the DUT.

Virtual networking components cannot leverage DUT model of testing as the DUT is not just the virtual device but includes the hardware components that were used to host the virtual device

Hence System Under Test (SUT) model MUST be used instead of the traditional device under test

With SUT model, the virtual networking component along with all software and hardware components that host the virtual networking component MUST be considered as part of the SUT.

Virtual networking components, because of their dependency on the underlying hardware and other software components, may end up leveraging NIC offload benefits, such as TCP Segmentation Offload (TSO), Large Receive Offload (LRO) and Rx / Tx Filters. Such underlying hardware and software level features, even though they may not be part of virtual networking stack itself, MUST be considered and documented. Note: Physical switches and routers, including the ones that act as initiators for NVOs, work with L2/L3 packets and may not be able to leverage TCP enhancements such as TSO.

Please refer to section 5 Figure 1 for a visual representation of System Under Test in the case of Intra-Host testing and section 5 Figure 2 for System Under Test in the case of Inter-Host testing.

3.2. Network Virtualization Platform

This document focuses on the Network Virtualization Overlay platform as outlined in RFC 8014 and use cases from RFC 8394.

Network Virtualization platforms, function closer to the application layer and are able to work with not only L2/L3 packets but also segments that leverage TCP optimizations such as Large Segment Offload (LSO).

NVPs leverage TCP stack optimizations such as TCP Segmentation Offload (TSO) and Large Receive Offload (LRO) that enables NVPs to work with much larger payloads of up to 64K unlike their counterparts such as NFVs.

Because of the difference in the payload, which translates into one operation per 64K of payload in NVP verses ~40 operations for the same amount of payload in NFV because of having to divide it to MTU sized packets, results in considerable difference in performance between NFV and NVP.

Please refer to figure 1 for a pictorial representation of this primary difference between NPV and NFV for a 64K payload segment/packet running on network set to 1500 bytes MTU.

Note: Payload sizes in figure 1 are approximates.

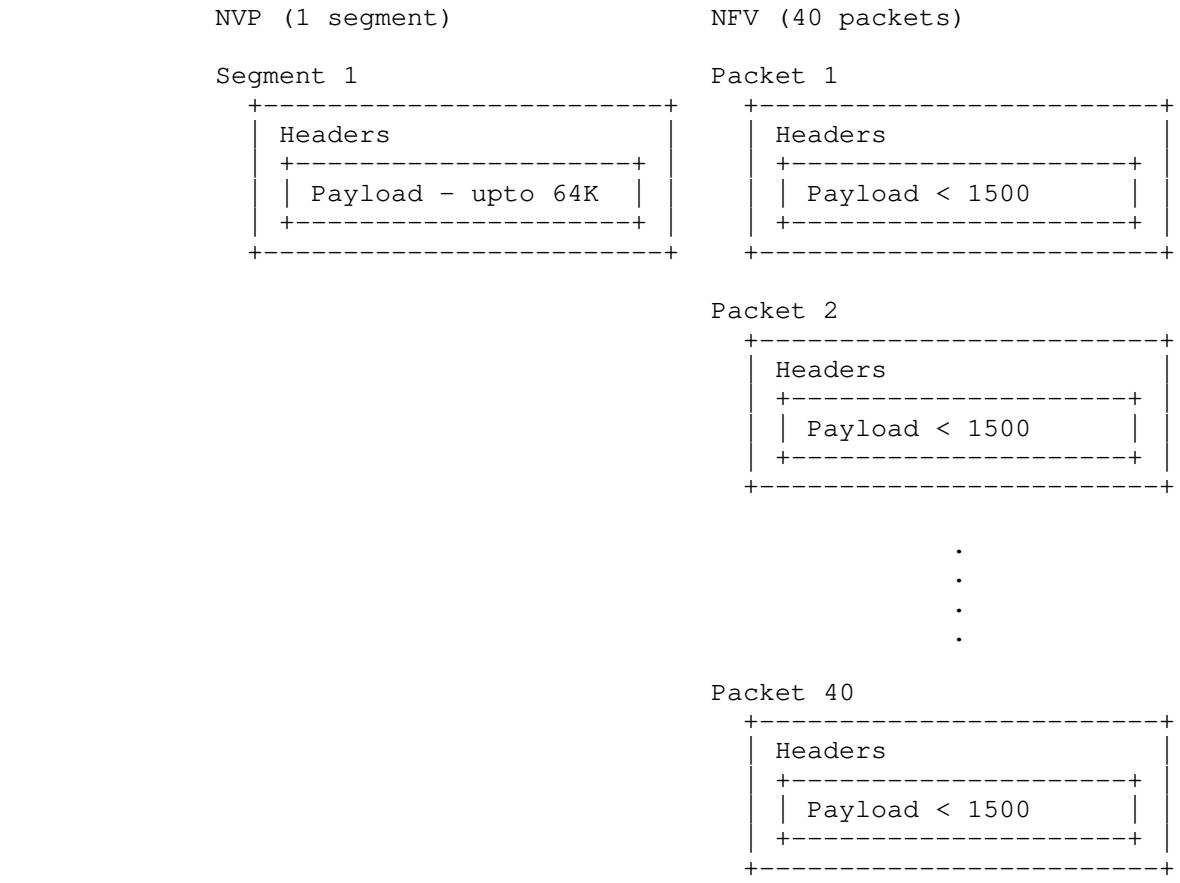

Figure 1! Payload NPV vs NFV

Hence, normal benchmarking methods are not relevant to the NVPs.

Instead, newer methods that leverage TCP optimizations MUST be used for testing Network Virtualization Platforms.

3.3. Microservices

Moving from traditional monolithic application architectures such as

the three tier web, app and db architectures to microservices model open up networking and security stacks to new scale and performance

related challenges. At a high level, in a microservices model, a traditional monolithic app that may use few IPs is broken down into 100s of individual one-responsibility-only applications where each application has connectivity and security related requirements. These 100s of small one-responsibility-only micro-services need their own IP and also secured into their own segments, hence pushing the scale boundaries of the overlay from both simple segmentation perspective and also from a security perspective.

For more details regarding microservices, please refer to wiki on microservices: https://en.wikipedia.org/wiki/Microservices

4. Scope

Focus of this document is the Network Virtualization Platform in two separate scenarios as outlined in RFC 8014 section 4, Network Virtualization Edge (NVE) and RFC 8394 section 1.1 Split-NVE and the associated learning phase:

4.1.1. Scenario 1

RFC 8014 Section 4.1 "NVE Co-located with server hypervisor": Where the entire NVE functionality will typically be implemented as part of the hypervisor and/or virtual switch on the server.

4.1.2. Scenario 2

RFC 8394 Section 1.1 "Split-NVE: A type of NVE (Network Virtualization Edge) where the functionalities are split across an end device supporting virtualization and an external network device."

4.1.3. Learning

Address learning rate is a key contributor to the overall performance of SUT specially in microservices type of use cases where a large amount of end-points are created and destroyed on demand.

4.1.4. Flow Optimization

There are several flow optimization algorithms that are designed to help improve latency or throughput. These optimizations MUST be documented.

4.1.5. Out of scope

This document does not address Network Function Virtualization which has been covered already by previous IETF documents

(https://datatracker.ietf.org/doc/draft-ietf-bmwg-virtual-net/?include_text=1).

Network Function Virtualization (NFV) focuses on being independent of networking hardware while providing the same functionality. In the case of NFV, traditional benchmarking methodologies recommended by IETF may be used. Considerations for Benchmarking Virtual Network Functions and Their Infrastructure IETF document addresses benchmarking NFVs.

Typical NFV implementations emulate in software, the characteristics and features of physical switches. They are similar to any physical L2/L3 switch from the perspective of the packet size, which is typically enforced based on the maximum transmission unit used.

4.2. Virtual Networking for Datacenter Applications

This document focuses on the virtual networking for east-west traffic within on-prem datacenter and/or cloud. For example, in a three tier app such web, app and db, this document focuses on the east-west traffic between web and app.

This document addresses scale requirements for modern application architectures such as Micro-services to consider whether the proposed solution is able to scale up to the demands of micro- services application models that basically have 100s of small services communicating on some standard ports such as http/https using protocols such as REST.

4.3. Interaction with Physical Devices

Virtual network components MUST NOT be tested independent of other components within the system. Example, unlike a physical router or a firewall, where the tests can be focused solely on the device, when testing a virtual router or firewall, multiple other devices may become part of the SUT. Hence the characteristics of these other traditional networking switches and routers, LB, FW etc. MUST be considered.

- Hashing method used
- Over-subscription rate
- Throughput available
- Latency characteristics

5. NVP Benchmarking Considerations

In virtual environments, the SUT may often share resources and resi
de
on the same physical hardware with other components involved in the tests. Hence SUT MUST be clearly documented. In these tests, a single hypervisor may host multiple servers, switches, routers, firewalls etc.

Intra host testing: Intra host testing helps in reducing the numbe
r
of components involved in a test. For example, intra host testing would help focus on the System Under Test, logical switch and the hardware that is running the hypervisor that hosts the logical switch, and eliminate other components. Because of the nature of virtual infrastructures and multiple elements being hosted on the same physical infrastructure, influence from other components canno
t
be completely ruled out. For example, unlike in physical infrastructures, logical routing or distributed firewall MUST NOT b
e
benchmarked independent of logical switching. System Under Test definition MUST include all components involved with that particula
r
test.

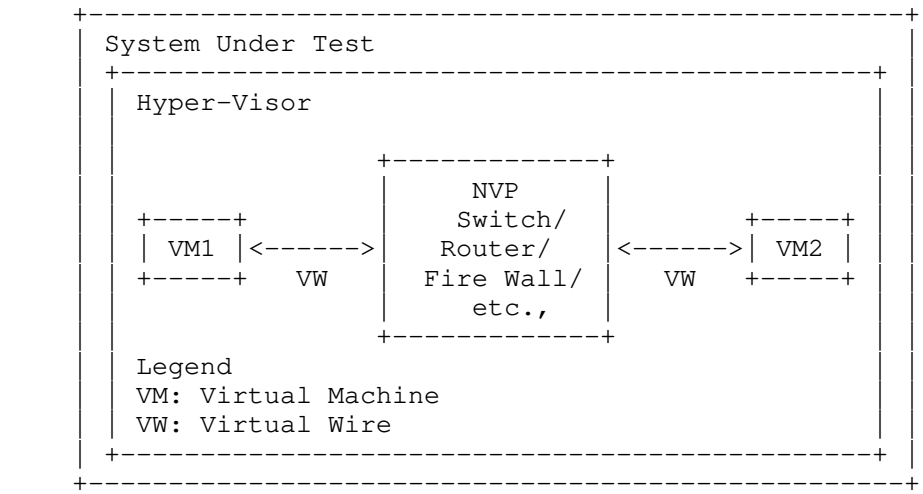

Figure 2! Intra-Host System Under Test

In the above figure, we only address the NVE co-located with the hypervisor.

Inter-host testing: Inter-host testing helps in profiling the underlying network interconnect performance. For example, when testing Logical Switching, inter host testing would not only test t
he

logical switch component but also any other devices that are part of the physical data center fabric that connects the two hypervisors. System Under Test MUST be well defined to help with repeatability of tests. System Under Test definition in the case of inter host testing, MUST include all components, including the underlying network fabric.

Figure 2 is a visual representation of system under test for inter-host testing.

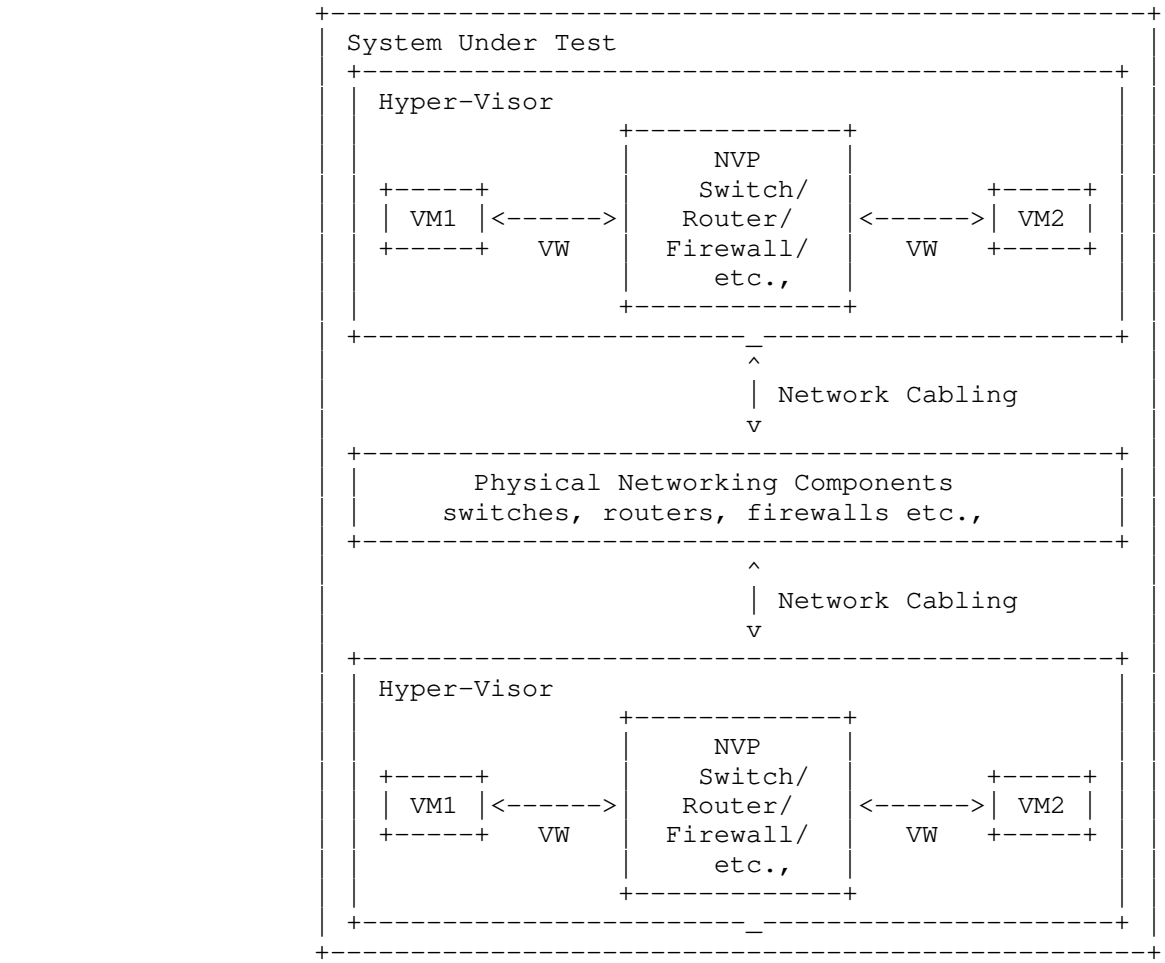

Figure 3! Inter-Host System Under Test

Virtual components have a direct dependency on the physical infrastructure that is hosting these resources. Hardware characteristics of the physical host impact the performance of the virtual components. The components that are being tested and the impact of the other hardware components within the hypervisor on th
e

performance of the SUT MUST be documented. Virtual component performance is influenced by the physical hardware components withi
n

the hypervisor. Access to various offloads such as TCP segmentation offload, may have significant impact on performance. Firmware and driver differences may also significantly impact results based on whether the specific driver leverages any hardware level offloads offered. Packet processing could be executed on shared or dedicated cores on the main processor or via a dedicated co-processor or embedded processor on NIC.

Hence, all physical components of the physical server running the hypervisor that hosts the virtual components MUST be documented along with the firmware and driver versions of all the components used to help ensure repeatability of test results. For example, BIOS configuration of the server MUST be documented as some of those changes are designed to improve performance. Please refer to Appendix A for a partial list of parameters to document.

5.1. Learning

SUT needs to learn all the addresses before running any tests. Address learning rate MUST be considered in the overall performance metrics because address learning rate has a high impact in microservices based use cases where there is huge churn of end points as they are created and destroyed on demand. In these cases, both the throughput at stable state, and the time taken to get to stable state MUST be tested and documented.

5.2. Traffic Flow Optimizations

Several mechanisms are employed to optimize traffic flows. Following are some examples:

5.2.1. Fast Path

A single flow may go through various switching, routing and firewalling decisions. While in the standard model, every single packet has to go through the entire process/pipeline, some optimizations help make this decision for the first packet, store the final state for that packet, and leverage it to skip the process for rest of the packets that are part of the same flow.

5.2.2. Dedicated cores / Co-processors

Packet processing is a CPU intensive workload. Some NVE's may use dedicated cores or a co-processor primarily for packet processing instead of sharing the cores used for the actual workloads. Such cases MUST be documented. Tests MUST be performed with both shared

and dedicated cores. Results and differences in results MUST be documented.

5.2.3. Prioritizing and de-prioritizing active flows

Certain algorithms may prioritize or de-prioritize traffic flows based on purely their network characteristics such as the length of the flow. For example, de-prioritize a long-lived flow. This coul
d
result in changing the performance of a flow over a period of time. Such optimizations MUST be documented, and tests MUST consist of long-lived flows to help capture the change in performance for such flows. Tests MUST note the point at which performance changes.

5.3. Server Architecture Considerations

When testing physical networking components, the approach taken is
to
consider the device as a black-box. With virtual infrastructure, this approach would no longer help as the virtual networking components are an intrinsic part of the hypervisor they are running on and are directly impacted by the server architecture used. Serve
r
hardware components define the capabilities of the virtual networki
ng
components. Hence, server architecture MUST be documented in detai
l
to help with repeatability of tests. And the entire hardware and software components become the SUT.

5.3.1. NVE Component considerations

5.3.1.1. NVE co-located

Components of NVE co-located may be hypervisor based or offloaded entirely to the NIC card or a hybrid model. In the case of hypervisor-based model, they may be running in user space or kernel space. Further, they may use dedicated cores, shared cores or in some cases dedicated co-processors. All the components and the process used MUST be documented.

5.3.1.2. NVE split

NVE split scenario generally has three primary components as documented per RFC 8394.

"tNVE: Terminal-side NVE. The portion of Split-NVE functionalitie
s
located on the end device supporting virtualization. The tNVE interacts with a Tenant System through an internal interface in the end device." tNVE may be made of either hypervisor controlled components such as hypervisor provided switches or NVE controlled

components where the network functionality is not provided by the hypervisor. In either case, the components used MUST be documented
.

"nNVE: Network-side NVE. The portion of Split-NVE functionalities located on the network device that is directly or indirectly connected to the end device that contains the corresponding NVE. Th
e
nNVE normally performs encapsulation to and decapsulation from the overlay network." All the functionality provided by the nNVE MUST
be
documented.

"External NVE: The physical network device that contains the nNVE.
"
Networking device hardware specs MUST be documented. Please use Apendix A for an example of the specs that MUST be documented.

In either case, NVE co-located or NVE split all the components MUST be documented. Where possible, individual components MUST be teste
d
independent of the entire system. For example, where possible, hypervisor provided switching functionality MUST be tested independent of the NVE.

Per RFC 8014, "For the split-NVE case, protocols will be needed tha
t
allow the hypervisor and NVE to negotiate and set up the necessary state so that traffic sent across the access link between a server and the NVE can be associated with the correct virtual network instance." Supported VM lifecycle events, from RFC 8394 section 2, MUST be documented as part of the benchmark process. This process MUST also include how the hypervisor and the external NVE have signaled each other to reach an agreement. Example, see section 2.
1
of RFC 8394 "VM creation event". The process used to update agreement status MUST also be documented.

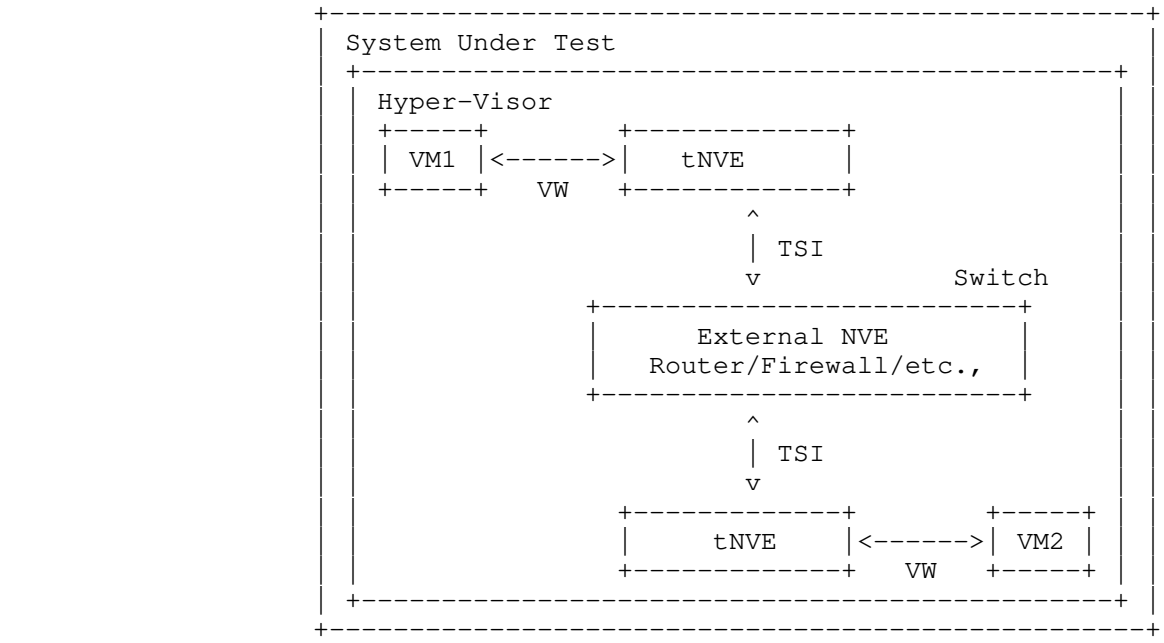

Figure 4 NVE Split collocated - System Under Test

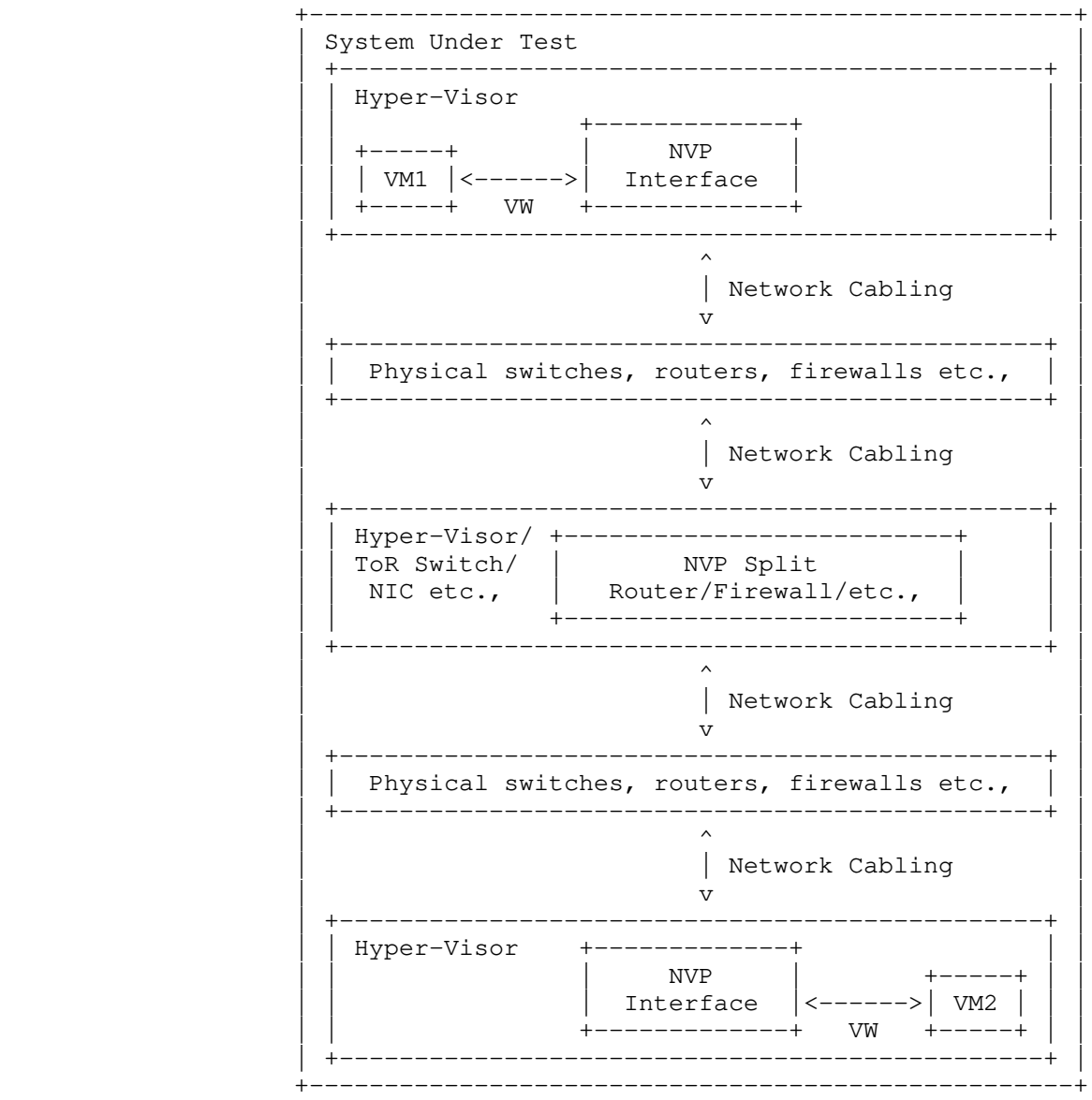

Figure 5 NVE Split not collocated - System Under Test

5.3.2. Frame format/sizes within the Hypervisor

Maximum Transmission Unit (MTU) limits physical network component's frame sizes. The most common max supported MTU for physical devices is 9000. However, 1500 MTU is the standard. Physical network testing and NFV uses these MTU sizes for testing. However, the virtual networking components that live inside a hypervisor, may work with much larger segments because of the availability of hardware and software based offloads. Hence, the normal smaller packets based testing is not relevant for performance testing of virtual networking components. All the TCP related configuration such as TSO size, number of RSS queues MUST be documented along with any other physical NIC related configuration.

NVE co-located may have a different performance profile when compared with NVE split because, the NVE co-located may have access to offloads that may not be available when the packet has to traverse the physical link. Such differences MUST be documented.

5.3.3. Baseline testing with Logical Switch

Logical switch is often an intrinsic component of the test system along with any other hardware and software components used for testing. Also, other logical components cannot be tested independent of the Logical Switch.

5.3.4. Repeatability

To ensure repeatability of the results, in the physical network component testing, much care is taken to ensure the tests are conducted with exactly the same parameters. Example parameters such as MAC addresses used.

When testing NVP components with an application layer test tool, there may be a number of components within the system that may not be available to tune or to ensure they maintain a desired state. Example: housekeeping functions of the underlying Operating System.

Hence, tests MUST be repeated a number of times and each test case MUST be run for at least 2 minutes if test tool provides such an option. Results SHOULD be derived from multiple test runs. Variance between the tests SHOULD be documented.

5.3.5. Tunnel encap/decap outside the Hypervisor

Logical network components may also have performance impact based on the functionality available within the physical fabric. Physical

fabric that supports NVO encap/decap is one such case that may have a different performance profile. Any such functionality that exists on the physical fabric MUST be part of the test result documentation to ensure repeatability of tests. In this case SUT MUST include the physical fabric if its being used for encap/decap operations.

5.3.6. SUT Hypervisor Profile

Physical networking equipment has well defined physical resource characteristics such as type and number of ASICs/SoCs used, amount of memory, type and number of processors etc., Virtual networking components performance is dependent on the physical hardware that hosts the hypervisor. Hence the physical hardware usage, which is part of SUT, for a given test MUST be documented, for example, CPU usage when running logical router.

CPU usage, changes based on the type of hardware available within the physical server. For example, TCP Segmentation Offload greatly reduces CPU usage by offloading the segmentation process to the NIC card on the sender side. Receive side scaling offers similar benefit on the receive side. Hence, availability and status of such hardware MUST be documented along with actual CPU/Memory usage when the virtual networking components have access to such offload capable hardware.

Following is a partial list of components that MUST be documented both in terms of what is available and also what is used by the SUT

- CPU - type, speed, available instruction sets (e.g. AES-NI)
- Memory - type, amount
- Storage - type, amount
- NIC Cards -
  - Type
  - number of ports
  - offloads available/used - following is a partial list of possible features
    - TCP Segmentation Offload
    - Large Receive Offload

- o Checksum Offloads
- o Receive Side Scaling
- o Other Queuing Mechanisms

  * drivers, firmware (if applicable)
  * HW revision

- o Libraries such as DPDK if available and used
- o Number and type of VMs used for testing and
  * vCPUs
  * RAM
  * Storage
  * Network Driver
  * Any prioritization of VM resources
  * Operating System type, version and kernel if applicable
  * TCP Configuration Changes - if any
  * MTU
- o Test tool
  * Workload type
  * Protocol being tested
  * Number of threads
  * Version of tool
- o For inter-hypervisor tests,
  * Physical network devices that are part of the test
    - o Note: For inter-hypervisor tests, system under test is no longer only the virtual component that is being

tested but the entire fabric that connects the virtual components become part of the system under test.

5.4. Benchmarking Tools Considerations

5.4.1. Considerations for NVE

Virtual network components in NVE work closer to the application layer then the physical networking components, which enables the virtual network components to take advantage of TCP optimizations such as TCP Segmentation Offload (TSO) and Large Receive Offload (LRO). Because of this optimizations, virtual network components work with type and size of segments that are often not the same type and size that the physical network works with. Hence, testing virtual network components MUST be done with application layer segments instead of the physical network layer packets. Testing MUST be done with application layer testing tools such as iperf, netperf etc.,

5.4.2. Considerations for Split-NVE

In the case of Split-NVE, since they may not leverage any TCP related optimizations, typical network test tools focused on packet processing MUST be used. However, the tools used MUST be able to leverage Receive Side Scaling (RSS).

6. Control Plane Scale Considerations

For a holistic approach to performance testing, control plane performance must also be considered. While the previous sections focused on performance tests after the SUT has come to a steady state, the following section focusses on tests to measure the time taken to bring the SUT to steady state.

In a physical network infrastructure world view, this could be various stages such as boot up time, time taken to apply configuration, BGP convergence time etc., In a virtual infrastructure world, this involves lot more components which may also be distributed across multiple hosts. Some of the components are:

- o VM Creation Event
- o VM Migration Event
- i How many total VMs can the SUT support

- What is the rate at which the SUT would allow creation of VMs

Please refer to section 2 of RFC 8394 for various VM events and their definitions. In the following section we further clarify some of the terms used in the above RFC.

VM Creation

For the purposes of NVP control plane testing, VM Creation event is when a VM starts participating for the first time on a NVP provided network. This involves various actions on the tNVE and NVP. Please refer to 2.1 "VM Creation Event" of RFC 8394 for more details.

In order to rule out any Hypervisor imposed limitations, System Under Test must first be profiled and baselined with-out the use of NVP components. For the purposes of baselining control plane, the VM used may have very small footprint such as DSL Linux which runs in 16MB RAM.

Once a baseline has been established for a single HV, a similar exercise MUST be done on multiple HVs to establish a baseline for the entire hypervisor domain. However, it may not be practical to have physical hosts and hence nested hosts may be used for this purpose

6.1.1. VM Events

Performance of various control plane activities which are associated with the System Under Test, MUST BE documented.

- VM Creation: Time taken to join the VMs to the SUT provided network
- Policy Realization: Time taken for policy realization on the VM
- VM Migration: Time taken to migrate a VM from one SUT provided network to another SUT provided network

For the test itself, the following process could be use:

1 API to call to join VM on the SUT provided network

2 Loop while incrementing a timer - till the VM comes online on the SUT provided network

Similarly, policy realization and VM migration may also be tested with a check on whether the VM is available or not available based on the type of policy that is applied.

6.1.2. Scale

SUT must also be tested to determine the maximum scale supported. Scale can be multi-faceted such as the following:

- Total # of VMs per Host
- Total # of VMs per one SUT Domain
- Total # of Hosts per one SUT Domain
- Total # of Logical Switches per one SUT Domain
    - Total # of VMs per one SUT provided Logical Switch
        - Per Host
        - Per SUT Domain
- Total # of Logical Routers per one SUT Domain
    - Total # of Logical Switches per one Logical Router
    - Total # of VMs on a single Logical Router
- Total # of Firewall Sections
- Total # of Firewall Rules per Section
- Total # of Firewall Rules applied per VM
- Total # of Firewall Rules applied per Host
- Total # of Firewall Rules per SUT

6.1.3. Control Plane Performance at Scale

Benchmarking MUST also test and document the control performance at scale. That is,

- Total # VMs that can be created in parallel
    - How long does the action take
- Total # of VMs that can be migrated in parallel
    - How long does the action take

- Total amount of time taken to apply 1 firewall across the entire VMs under a SUT
- Time taken to apply 1000s rules on a SUT

7. Security Considerations

Benchmarking activities as described in this memo are limited to technology characterization of a Device Under Test/System Under Test (DUT/SUT) using controlled stimuli in a laboratory environment, with dedicated address space and the constraints specified in the sections above.

The benchmarking network topology will be an independent test setup and MUST NOT be connected to devices that may forward the test traffic into a production network, or misroute traffic to the test management network.

Further, benchmarking is performed on a 'black-box' basis, relying solely on measurements observable external to the DUT/SUT.

Special capabilities SHOULD NOT exist in the DUT/SUT specifically for benchmarking purposes. Any implications for network security arising from the DUT/SUT SHOULD be identical in the lab and in production networks.

8. IANA Considerations

No IANA Action is requested at this time.

9. Conclusions

Network Virtualization Platforms, because of their proximity to the application layer and since they can take advantage of TCP stack optimizations, do not function on packets/sec basis. Hence, traditional benchmarking methods, while still relevant for Network Function Virtualization, are not designed to test Network Virtualization Platforms. Also, advances in application architectures such as micro-services, bring new challenges and need benchmarking not just around throughput and latency but also around scale. New benchmarking methods that are designed to take advantage of the TCP optimizations or needed to accurately benchmark performance of the Network Virtualization Platforms

10. References

10.1. Normative References

[RFC7364] T. Narten, E. Gray, D. Black, L. Fang, L. Kreeger, M. Napierala, 'Problem Statement: Overlays for Network Virtualization', RFC 7364, October 2014, https://datatracker.ietf.org/doc/rfc7364/

[RFC8014] D. Black, J. Hudson, L. Kreeger, M. Lasserre, T. Narten 'An Architecture for Data-Center Network Virtualization over Layer 3 (NVO3)', RFC 8014, December 2016, https://tools.ietf.org/html/rfc8014

[RFC8394] Y. Li, D. Eastlake 3rd, L. Kreeger, T. Narten, D. Black 'Split Network Virtualization Edge (Split-NVE) Control-Plane Requirements', RFC 8394, May 2018, https://tools.ietf.org/html/rfc8394

[nv03] IETF, WG, Network Virtualization Overlays, <https://datatracker.ietf.org/wg/nvo3/documents/>

10.2. Informative References

[RFC8172] A. Morton 'Considerations for Benchmarking Virtual Network Functions and Their Infrastructure', RFC 8172, July 2017, https://tools.ietf.org/html/rfc8172

11. Acknowledgments

This document was prepared using 2-Word-v2.0.template.dot.

Appendix A.!Partial List of Parameters to Document

A.1. CPU

CPU Vendor

CPU Number

CPU Architecture

# of Sockets (CPUs)

# of Cores

Clock Speed (GHz)

Max Turbo Freq. (GHz)

Cache per CPU (MB)

# of Memory Channels

Chipset

Hyperthreading (BIOS Setting)

Power Management (BIOS Setting)

VT-d

Shared vs Dedicated packet processing

User space vs Kernel space packet processing

A.2. Memory

Memory Speed (MHz)

DIMM Capacity (GB)

# of DIMMs

DIMM configuration

Total DRAM (GB)

A.3. NIC

Vendor

Model

Port Speed (Gbps)

Ports

PCIe Version

PCIe Lanes

Bonded

Bonding Driver

Kernel Module Name

Driver Version

VXLAN TSO Capable

VXLAN RSS Capable

Ring Buffer Size RX

Ring Buffer Size TX

A.4. Hypervisor

Hypervisor Name

Version/Build

Based on

Hotfixes/Patches

OVS Version/Build

IRQ balancing

vCPUs per VM

Modifications to HV

Modifications to HV TCP stack

Number of VMs

IP MTU

Flow control TX (send pause)

Flow control RX (honor pause)

Encapsulation Type

A.5. Guest VM

Guest OS & Version

Modifications to VM

IP MTU Guest VM (Bytes)

Test tool used

Number of NetPerf Instances

Total Number of Streams

Guest RAM (GB)

A.6. Overlay Network Physical Fabric

Vendor

Model

and Type of Ports

Software Release

Interface Configuration

Interface/Ethernet MTU (Bytes)

Flow control TX (send pause)

Flow control RX (honor pause)

A.7. Gateway Network Physical Fabric

Vendor

Model

and Type of Ports

Software Release

Interface Configuration

Interface/Ethernet MTU (Bytes)

Flow control TX (send pause)

Flow control RX (honor pause)

A.8. Metrics

Drops on the virtual infrastructure

Drops on the physical underlay infrastructure

Authors' Addresses

Samuel Kommu
VMware
3401 Hillview Ave
Palo Alto, CA, 94304

Email: skommu@vmware.com

Jacob Rapp
VMware
3401 Hillview Ave
Palo Alto, CA, 94304

Email: jrapp@vmware.com