NSX Operations Guide

Rev. 1.5

March 2016

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

Version History

- Document version 1.0, June 19, 2015
- Document version 1.1, June 26, 2015
- Document version 1.2, August 4, 2015
- Document version 1.3, August 7, 2015
- Document version 1.4, October 22, 2015 (Added log rotation schedule information.)
- Document version 1.5, March 23, 2016(Made changes to backup/restore.)

Contents

Introduction

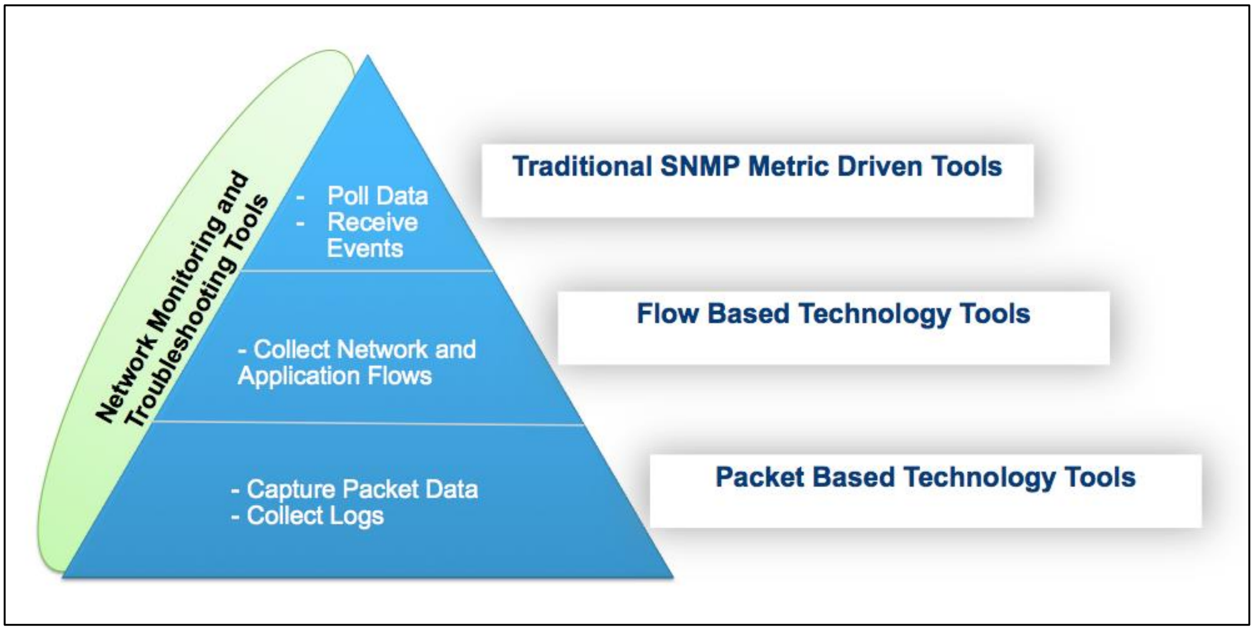
Purpose

This guide shows how to perform day-to-day management of an NSX for vSphere ("NSX-v") deployment. This information can be used to help plan and carry out operational monitoring and management of your NSX-v implementation.

To monitor physical network operations, administrators have traditionally collected various types of data from the devices that provide network connectivity and services. Broadly the data can be categorized as:

- Statistics and events
- Flow level data
- Packet level data

Monitoring and troubleshooting tools use the above types of data and help administrators manage and operate networks. Collectively, these types of information are referred to as “network and performance monitoring and diagnostics” (NPMD) data. The diagram below summarizes the types of NPMD data and the tools that consume this information.


Network Monitoring and Troubleshooting Tools
- Poll Data
- Receive Events
- Collect Network and Application Flows
- Capture Packet Data
- Collect Logs
Traditional SNMP Metric Driven Tools
Flow Based Technology Tools
Packet Based Technology Tools

The tools used for monitoring physical networks can be used to monitor virtual networks as well. Using standard protocols, the NSX platform provides network monitoring data similar to that provided by physical devices, giving administrators a clear view of virtual network conditions.

In this document, we'll describe how an administrator can monitor and retrieve network statistics, network flow information, packet information, and NSX system events.

Audience

This document is intended for those involved in the configuration, maintenance, and administration of VMware NSX-v. The intended audience includes the following business roles:

- Architects and planners responsible for driving architecture-level decisions.

- Security decision makers responsible for business continuity planning.

- Consultants, partners, and IT personnel, who need the knowledge for deploying the solution

This guide is written with the assumption that an administrator who will use these procedures is familiar with VMware vSphere and NSX-v, and we assume the reader has as strong networking background. For detailed explanations of NSX-v concepts and terminology, please refer to the NSX for vSphere documentation at https://www.vmware.com/support/pubs/nsx_pubs.html and the *VMware NSX for vSphere Network Virtualization Design Guide*, available from https://communities.vmware.com/community/vmtn/nsx.

Scope

This guide covers NSX-v and its integration with core VMware technologies such as vSphere and Virtual Distributed Switch (vDS). It does not attempt to cover architectural design decisions or installation. Also, while there are third-party integrations and extensive APIs available to programmatically program and manage NSX, this document does not focus on APIs or third-party integration including other VMware products. We do mention specific APIs when they offer a recommended or efficient method for configuring NSX, and when there is no direct UI function available to perform the desired action.

Monitoring NSX

Statistics

NSX provides API, CLI, and user interface tools to monitor various network components. These tools allow you to build dashboards that monitor activity and highlight abnormalities.

SNMP Monitoring of Virtual Switch Statistics

To monitor VMware Distributed Switch (VDS) using SNMP, the following steps must be performed:

1. Enable SNMP agent on each ESXi host where the VDS is configured - Refer to this document on how to configure SNMP v1/v2c/v3 on your ESXi host - http://pubs.vmware.com/vsphere-55/index.jsp#com.vmware.vsphere.monitoring.doc/GUID-8EF36D7D-59B6-4C74-B1AA-4A9D18AB6250.html
2. Load SNMP MIBs in your SNMP Management station
3. Monitor SNMP MIBs

This section will cover the Part II and Part III in more details.

Load SNMP MIBs in your SNMP Management station

VMware Distributed Switch supports various MIBs for monitoring the ESXi host. The following MIBs are of interest for monitoring traffic for Virtual Machines, VMKernel NICs and Physical NICs - IF-MIB, IEEE8021-Q-BRIDGE-MIB, IEEE8021-BRIDGE-MIB. To load these MIBs in your SNMP Management Station, you are required to load additional MIBs to satisfy the dependency. All the required SNMP MIBs can be downloaded as a single zip file by visiting “MY VMware” site at https://my.vmware.com → Downloads and searching for SNMP MIBs (and version of vSphere).

Sample URL for downloading SNMP MIBs for vSphere 5.5 -
https://my.vmware.com/web/vmware/details?downloadGroup=SNMPMIBS550&productId=408

Unzip the .zip file and then Load the listed MIBs in the following order to satisfy dependencies:

- SNMPv2-SMI
- SNMPv2-TC
- SNMP-FRAMEWORK-MIB
- SNMPv2-CONF
- SNMPv2-MIB
- IANAifType-MIB
- IF-MIB
- IEEE8021-TC-MIB
- IEEE8021-BRIDGE-MIB

- BRIDGE-MIB
- P-BRIDGE-MIB
- RMON-MIB
- TOKEN-RING-RMON-MIB
- RMON2-MIB
- Q-BRIDGE-MIB
- IEEE8021-Q-BRIDGE-MIB
- VMWARE-ROOT-MIB
- VMWARE-PRODUCTS-MIB
- VMWARE-TC-MIB
- VMWARE-ENV-MIB
- VMWARE-VMINFO-MIB

Monitor VM Traffic statistics

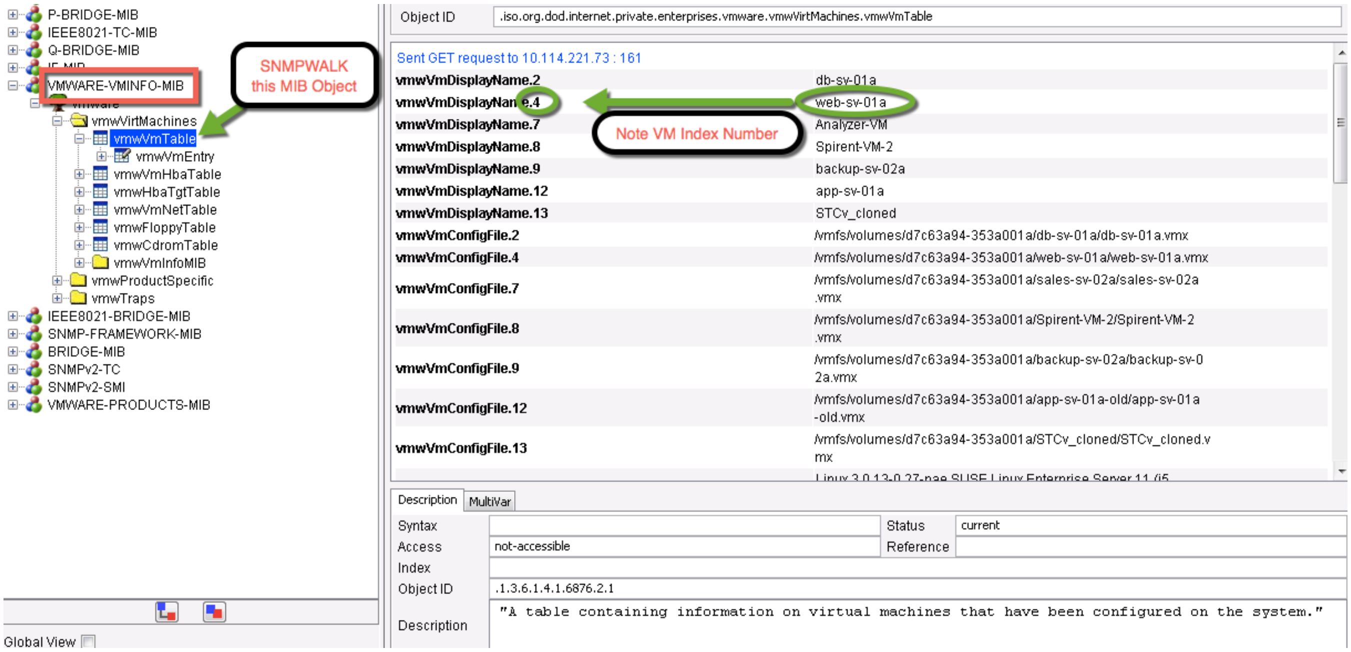
IEEE8021-Q-BRIDGE-MIB provides statistics for ports connected to a bridge/switch. In context of VDS, this MIB will provide traffic statistics for a virtual switch port on the VDS.

In order to gather statistics for a VM, the first step is to identify the mac address of the Virtual Machine. Using this MAC Address as the identifier, we will find the Port ID on the VDS and then use this Port ID to gather the statistics.

Finding MAC Address of the Virtual Machine's vNIC.

The information of the Virtual Machine is obtained from the **VMWARE-VMINFO-MIB**. Perform **SNMPWALK** on the MIB Object – "**.iso.org.dod.internet.private.enterprises.vmware.vmwVirtMachines.vmwVmTable**" as shown in the below figure.

Identify the VM that you would like to monitor using the **vmwVmDisplayName** object. In our case, we will monitor the VM "**web-sv-01a**". Note the number VM Index "**4**" corresponding to this VM.

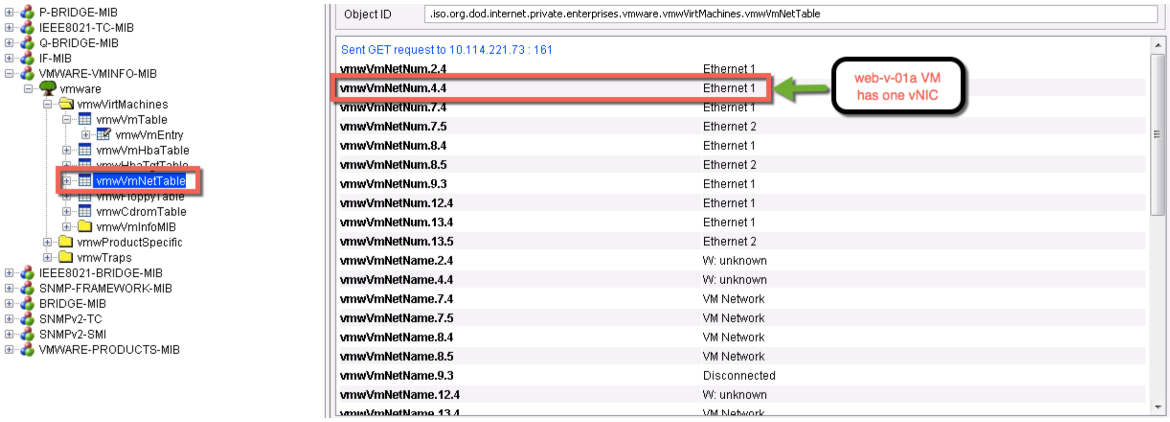


Object ID .iso.org.dod.internet.private.enterprises.vmware.vmwVirtMachines.vmwVmTable

```
Sent GET request to 10.114.221.73 : 161
vmwVmDisplayName.2          db-sv-01a
vmwVmDisplayName.4          web-sv-01a
vmwVmDisplayName.7          Analyzer-VM
vmwVmDisplayName.8          Spirent-VM-2
vmwVmDisplayName.9          backup-sv-02a
vmwVmDisplayName.12         app-sv-01a
vmwVmDisplayName.13         STCv_cloned
vmwVmConfigFile.2           /vmfs/volumes/d7c63a94-353a001a/db-sv-01a/db-sv-01a.vmx
vmwVmConfigFile.4           /vmfs/volumes/d7c63a94-353a001a/web-sv-01a/web-sv-01a.vmx
vmwVmConfigFile.7           /vmfs/volumes/d7c63a94-353a001a/sales-sv-02a/sales-sv-02a
                            .vmx
vmwVmConfigFile.8           /vmfs/volumes/d7c63a94-353a001a/Spirent-VM-2/Spirent-VM-2
                            .vmx
vmwVmConfigFile.9           /vmfs/volumes/d7c63a94-353a001a/backup-sv-02a/backup-sv-0
                            2a.vmx
vmwVmConfigFile.12          /vmfs/volumes/d7c63a94-353a001a/app-sv-01a-old/app-sv-01a
                            -old.vmx
vmwVmConfigFile.13          /vmfs/volumes/d7c63a94-353a001a/STCv_cloned/STCv_cloned.v
                            mx
                            Linux 3.0.13-0.27-pae SUSE Linux Enterprise Server 11 (i5
```

Syntax		Status	current
Access	not-accessible	Reference	
Index			
Object ID	.1.3.6.1.4.1.6876.2.1		
Description	"A table containing information on virtual machines that have been configured on the system."		

In the next step, we will identify the interface on the VM that we would like to monitor. Perform **SNMPWALK** on the MIB Object – "**.iso.org.dod.internet.private.enterprises.vmware.vmwVirtMachines.vmwVmNetTable**" as shown in the below figure. Object **vmwVmNetNum.[VmNetVmIdx].[VmNetIdx]** will provide details about the vNIC of the VM. In our example, the object **vmwVmNetNum.4.4** represent the single vNIC of this VM.



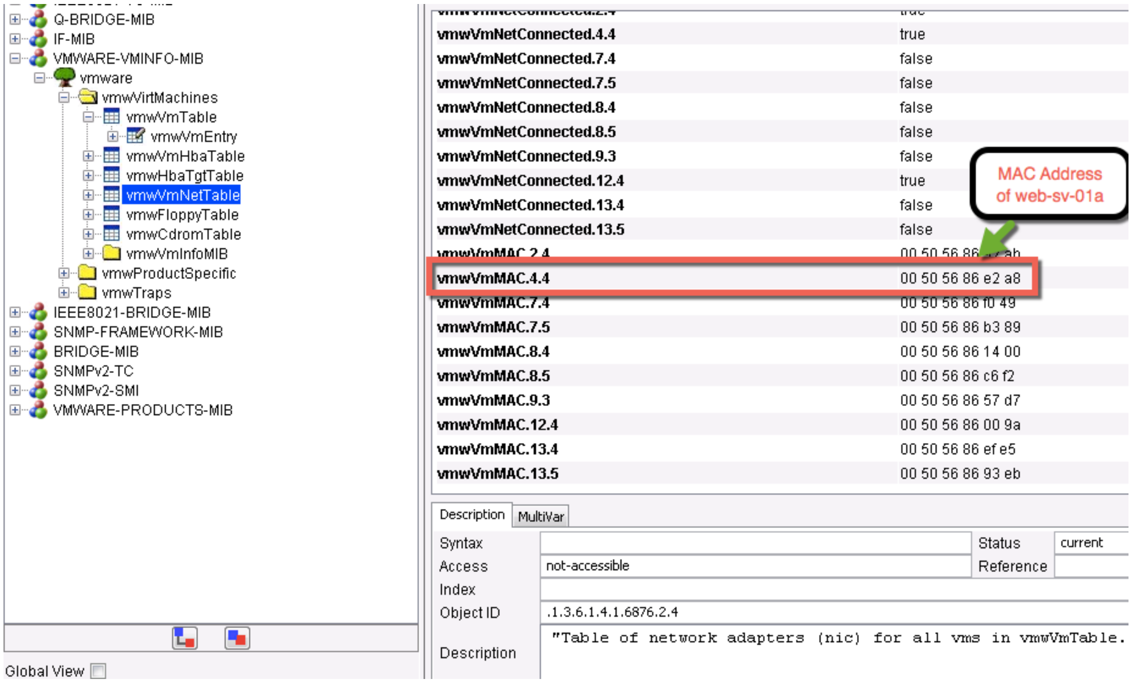
Object ID .iso.org.dod.internet.private.enterprises.vmware.vmwVirtMachines.vmwVmNetTable

```
Sent GET request to 10.114.221.73 : 161
vmwVmNetNum.2.4             Ethernet 1
vmwVmNetNum.4.4             Ethernet 1
vmwVmNetNum.7.4             Ethernet 1
vmwVmNetNum.7.5             Ethernet 2
vmwVmNetNum.8.4             Ethernet 1
vmwVmNetNum.8.5             Ethernet 2
vmwVmNetNum.9.3             Ethernet 1
vmwVmNetNum.12.4            Ethernet 1
vmwVmNetNum.13.4            Ethernet 1
vmwVmNetNum.13.5            Ethernet 2
vmwVmNetName.2.4            W: unknown
vmwVmNetName.4.4            W: unknown
vmwVmNetName.7.4            VM Network
vmwVmNetName.7.5            VM Network
vmwVmNetName.8.4            VM Network
vmwVmNetName.8.5            VM Network
vmwVmNetName.9.3            Disconnected
vmwVmNetName.12.4           W: unknown
vmwVmNetName.13.4           VM Network
```

Note: If the Virtual Machine has more than one vNIC, then the last digit of this object [**VmNetIdx]** will increment to represent additional vNICs.

[As a side example, you can verify that VM **– “STCv_cloned”** (represented by number 13) has two vNIC represented by **vmwVmNetNum.13.4 and vmwVmNetNum.13.5]**

Back to our example, we will now find the mac address corresponding to vNIC of **web-sv-01a**.



vmwVmNetConnected.2.4	true
vmwVmNetConnected.4.4	true
vmwVmNetConnected.7.4	false
vmwVmNetConnected.7.5	false
vmwVmNetConnected.8.4	false
vmwVmNetConnected.8.5	false
vmwVmNetConnected.9.3	false
vmwVmNetConnected.12.4	true
vmwVmNetConnected.13.4	false
vmwVmNetConnected.13.5	false
vmwVmMAC.2.4	00 50 56 86 [illegible] ab
vmwVmMAC.4.4	00 50 56 86 e2 a8
vmwVmMAC.7.4	00 50 56 86 f0 49
vmwVmMAC.7.5	00 50 56 86 b3 89
vmwVmMAC.8.4	00 50 56 86 14 00
vmwVmMAC.8.5	00 50 56 86 c6 f2
vmwVmMAC.9.3	00 50 56 86 57 d7
vmwVmMAC.12.4	00 50 56 86 00 9a
vmwVmMAC.13.4	00 50 56 86 ef e5
vmwVmMAC.13.5	00 50 56 86 93 eb

Syntax: | Status: current
Access: not-accessible | Reference:
Index:
Object ID: .1.3.6.1.4.1.6876.2.4
Description: "Table of network adapters (nic) for all vms in vmwVmTable.

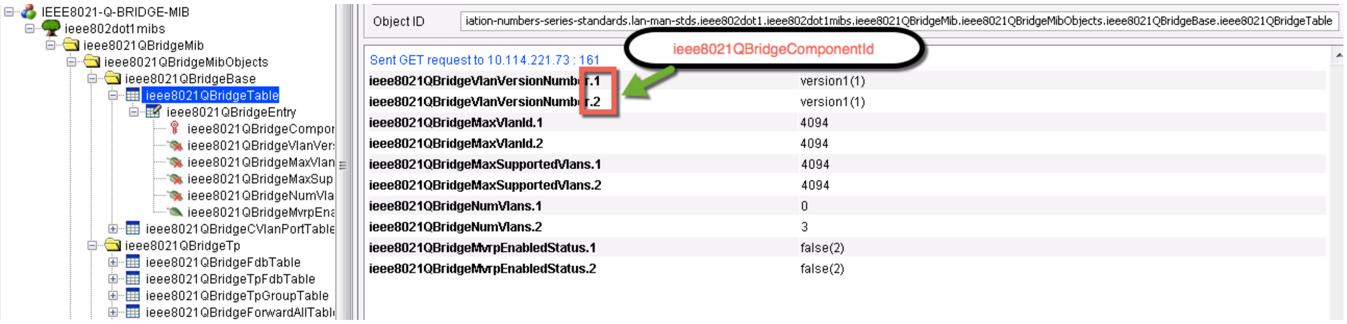
Finding port number on VDS on the host.

In this step, we will query the **IEEE8021-Q-BRIDGE-MIB** to find the VDS.

Step a) There may be multiple bridge instance on the ESXi host. It is very common to have a mix of VSS (one+) and VDS (>=1) on the host. Each of these switch instances will have a Bridge Entry in the MIB db. To identify the correct bridge instance we will have to circle through each bridge table until we find the correct bridge.

To list all the bridge instances on the host, SNMPWALK the object “IEE8021-Q-BRIDGE-MIB::iee8021QBridgeTable”.

The “ieee8021QBridgeComponentId” increments for each instance of bridge starting with default value of 1 when there is presence of at least one instance. In our screenshot below, we see there are there are two instances of virtual switches on our host.

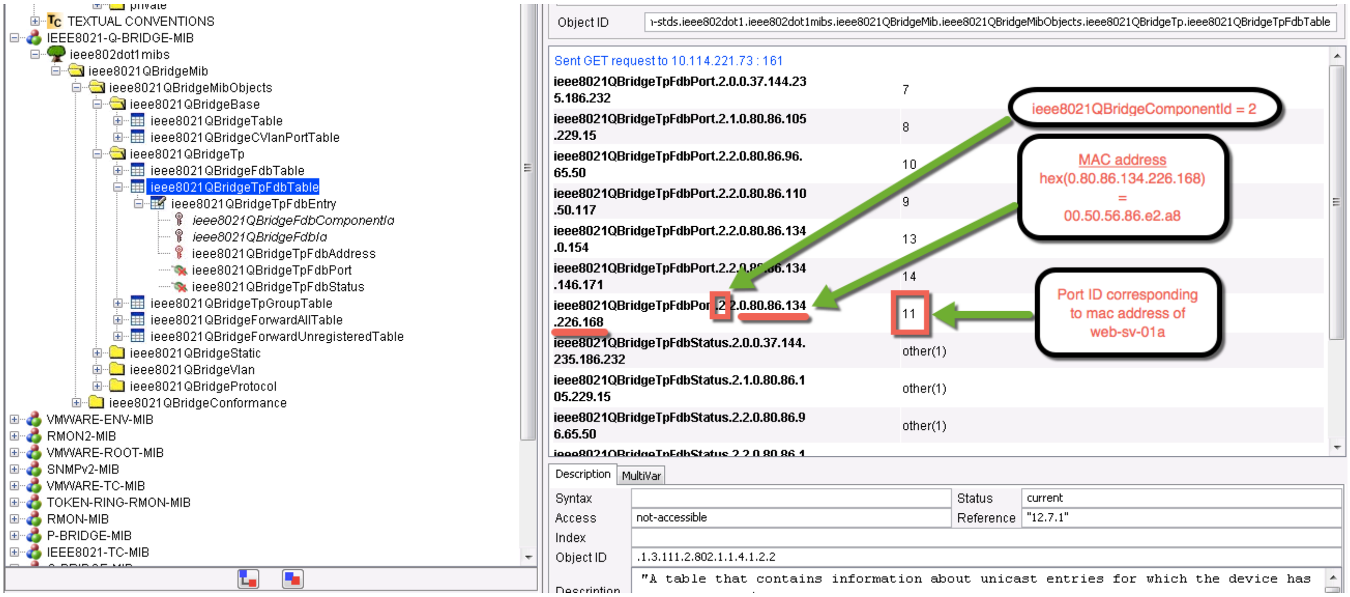


Object ID iation-numbers-series-standards.lan-man-stds.ieee802dot1.ieee802dot1mibs.ieee8021QBridgeMib.ieee8021QBridgeMibObjects.ieee8021QBridgeBase.ieee8021QBridgeTable

```
Sent GET request to 10.114.221.73 : 161
ieee8021QBridgeVlanVersionNumber.1          version1(1)
ieee8021QBridgeVlanVersionNumber.2          version1(1)
ieee8021QBridgeMaxVlanId.1                  4094
ieee8021QBridgeMaxVlanId.2                  4094
ieee8021QBridgeMaxSupportedVlans.1          4094
ieee8021QBridgeMaxSupportedVlans.2          4094
ieee8021QBridgeNumVlans.1                   0
ieee8021QBridgeNumVlans.2                   3
ieee8021QBridgeMvrpEnabledStatus.1          false(2)
ieee8021QBridgeMvrpEnabledStatus.2          false(2)
```

Now we will find the port number where our VM's mac address is present. We will query the "IEEE8021-Q-BRIDGE-MIB::ieee8021QBridgeTpFdbTable" Object.

We see that the mac address (in hex) – 00.50.56.86.e2.a8 is observed on Bridge Instance with ID = 2 and the Port ID = 11. The Object ieee8021QBridgeTpFdbAddress represents mac address in dotted decimal format. Decimal String (00.50.56.86.e2.a8) = 0.80.134.226.168



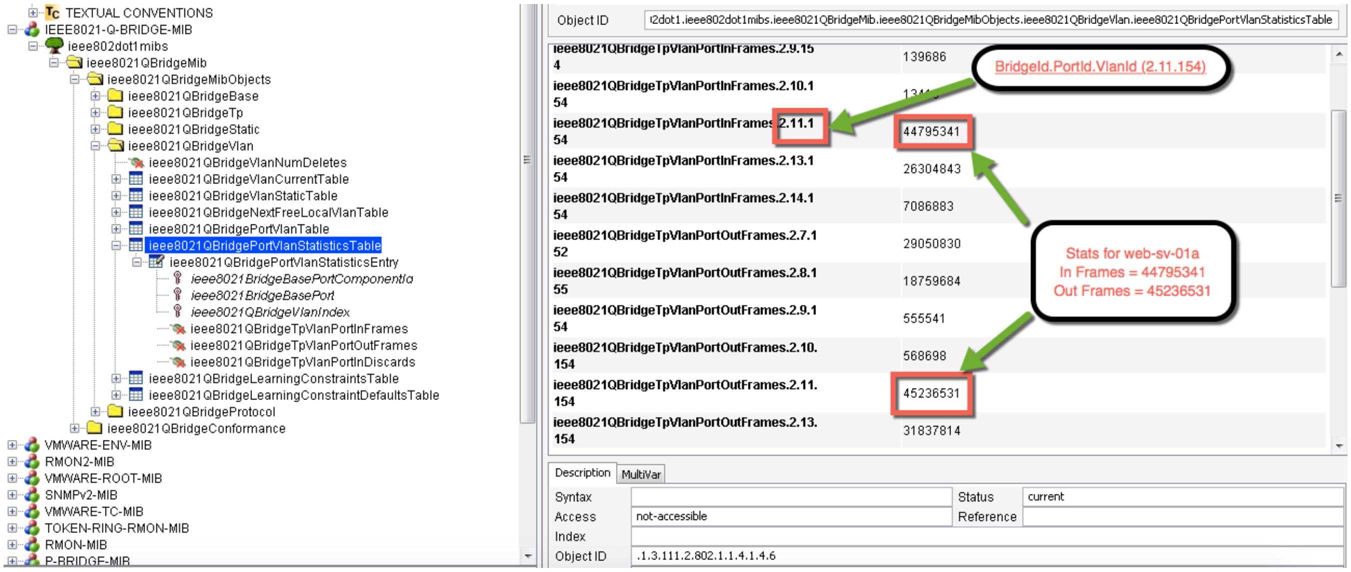
Object ID ı-stds.ieee802dot1.ieee802dot1mibs.ieee8021QBridgeMib.ieee8021QBridgeMibObjects.ieee8021QBridgeTp.ieee8021QBridgeTpFdbTable

```
Sent GET request to 10.114.221.73 : 161
ieee8021QBridgeTpFdbPort.2.0.0.37.144.23
5.186.232                                   7
ieee8021QBridgeTpFdbPort.2.1.0.80.86.105
.229.15                                     8
ieee8021QBridgeTpFdbPort.2.2.0.80.86.96.
65.50                                       10
ieee8021QBridgeTpFdbPort.2.2.0.80.86.110
.50.117                                     9
ieee8021QBridgeTpFdbPort.2.2.0.80.86.134
.0.154                                      13
ieee8021QBridgeTpFdbPort.2.2.0.80.86.134
.146.171                                    14
ieee8021QBridgeTpFdbPort.2.2.0.80.86.134
.226.168                                    11
ieee8021QBridgeTpFdbStatus.2.0.0.37.144.
235.186.232                                 other(1)
ieee8021QBridgeTpFdbStatus.2.1.0.80.86.1
05.229.15                                   other(1)
ieee8021QBridgeTpFdbStatus.2.2.0.80.86.9
6.65.50                                     other(1)
ieee8021QBridgeTpFdbStatus.2.2.0.80.86.1
```

Syntax		Status	current
Access	not-accessible	Reference	"12.7.1"
Index			
Object ID	.1.3.111.2.802.1.1.4.1.2.2		
Description	"A table that contains information about unicast entries for which the device has		

Finally we will poll the IEEE8021-Q-BRIDGE-MIB::ieee8021QBridgePortVlanStatisticsTable to gather In Frame, Out Frame, In Discards statistics.

ieee8021QBridgeTpVlanPortInFrames, ieee8021QBridgeTpVlanPortOutFrames and ieee8021QBridgeTpVlanPortInDiscards provides the statistics for the given Port on a Bridge instance. In our example, ieee8021QBridgeTpVlanPortInFrames.2.11.154 provides statistics for Port 11 on Bridge ID = 2. Also note the statistics are reported per VLAN. Since this port is an access port assigned to VLAN 154, we observe statistics only for that single VLAN. If this port were a .1Q trunk, then the statistics would be reported individually for each VLAN separately.



```
⊞ TEXTUAL CONVENTIONS
⊟ IEEE8021-Q-BRIDGE-MIB
    ⊟ ieee802dot1mibs
        ⊟ ieee8021QBridgeMib
            ⊟ ieee8021QBridgeMibObjects
                ⊞ ieee8021QBridgeBase
                ⊞ ieee8021QBridgeTp
                ⊞ ieee8021QBridgeStatic
                ⊟ ieee8021QBridgeVlan
                    ieee8021QBridgeVlanNumDeletes
                    ⊞ ieee8021QBridgeVlanCurrentTable
                    ⊞ ieee8021QBridgeVlanStaticTable
                    ⊞ ieee8021QBridgeNextFreeLocalVlanTable
                    ⊞ ieee8021QBridgePortVlanTable
                    ⊟ ieee8021QBridgePortVlanStatisticsTable
                        ⊟ ieee8021QBridgePortVlanStatisticsEntry
                            ieee8021BridgeBasePortComponentId
                            ieee8021BridgeBasePort
                            ieee8021QBridgeVlanIndex
                            ieee8021QBridgeTpVlanPortInFrames
                            ieee8021QBridgeTpVlanPortOutFrames
                            ieee8021QBridgeTpVlanPortInDiscards
                    ⊞ ieee8021QBridgeLearningConstraintsTable
                    ⊞ ieee8021QBridgeLearningConstraintDefaultsTable
                ⊞ ieee8021QBridgeProtocol
            ⊞ ieee8021QBridgeConformance
⊞ VMWARE-ENV-MIB
⊞ RMON2-MIB
⊞ VMWARE-ROOT-MIB
⊞ SNMPv2-MIB
⊞ VMWARE-TC-MIB
⊞ TOKEN-RING-RMON-MIB
⊞ RMON-MIB
⊞ P-BRIDGE-MIB
```

Object ID: l2dot1.ieee802dot1mibs.ieee8021QBridgeMib.ieee8021QBridgeMibObjects.ieee8021QBridgeVlan.ieee8021QBridgePortVlanStatisticsTable

ieee8021QBridgeTpVlanPortInFrames.2.9.154	139686
ieee8021QBridgeTpVlanPortInFrames.2.10.154	1344
ieee8021QBridgeTpVlanPortInFrames.2.11.154	44795341
ieee8021QBridgeTpVlanPortInFrames.2.13.154	26304843
ieee8021QBridgeTpVlanPortInFrames.2.14.154	7086883
ieee8021QBridgeTpVlanPortOutFrames.2.7.152	29050830
ieee8021QBridgeTpVlanPortOutFrames.2.8.155	18759684
ieee8021QBridgeTpVlanPortOutFrames.2.9.154	555541
ieee8021QBridgeTpVlanPortOutFrames.2.10.154	568698
ieee8021QBridgeTpVlanPortOutFrames.2.11.154	45236531
ieee8021QBridgeTpVlanPortOutFrames.2.13.154	31837814

BridgeId.PortId.VlanId (2.11.154)

Stats for web-sv-01a
In Frames = 44795341
Out Frames = 45236531

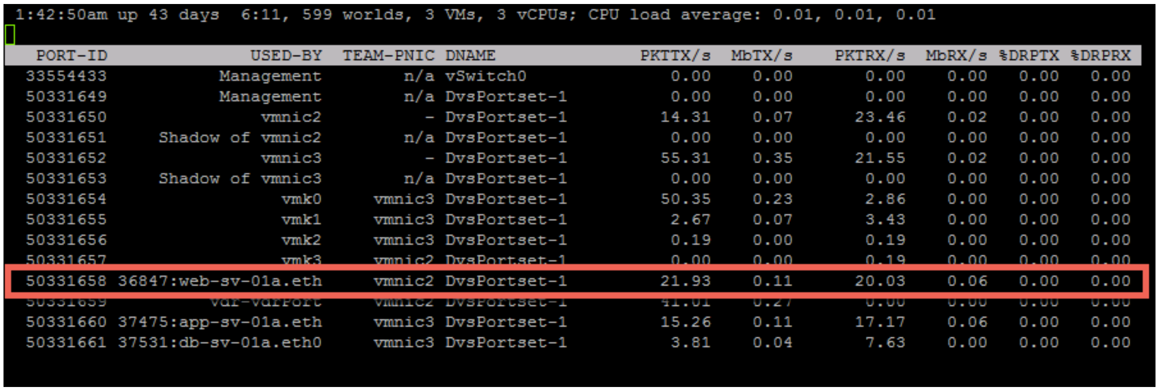
Syntax		Status	current
Access	not-accessible	Reference	
Index			
Object ID	.1.3.111.2.802.1.1.4.1.4.6		

Command Line Monitoring of Virtual Switch Statistics

We can monitor vSwitch statistics in real time such as packets/s, bytes/s, and dropped packets % using the “esxtop” command.

Finally we will poll the IEEE8021-Q-BRIDGE-MIB::ieee8021QBridgePortVlanStatisticsTable to gather In Frame, Out Frame, In Discards statistics.

Run **esxtop** and then press “n” to change the view for network statistics. This view provides statistics for all the instances of vswitches that are present on the host, including VMware Distributed Switch and VMware Standard Switch.



```
 1:42:50am up 43 days  6:11, 599 worlds, 3 VMs, 3 vCPUs; CPU load average: 0.01, 0.01, 0.01

   PORT-ID              USED-BY  TEAM-PNIC DNAME              PKTTX/s  MbTX/s    PKTRX/s  MbRX/s %DRPTX %DRPRX
  33554433           Management        n/a vSwitch0              0.00    0.00       0.00    0.00   0.00   0.00
  50331649           Management        n/a DvsPortset-1          0.00    0.00       0.00    0.00   0.00   0.00
  50331650               vmnic2          - DvsPortset-1         14.31    0.07      23.46    0.02   0.00   0.00
  50331651     Shadow of vmnic2        n/a DvsPortset-1          0.00    0.00       0.00    0.00   0.00   0.00
  50331652               vmnic3          - DvsPortset-1         55.31    0.35      21.55    0.02   0.00   0.00
  50331653     Shadow of vmnic3        n/a DvsPortset-1          0.00    0.00       0.00    0.00   0.00   0.00
  50331654                 vmk0     vmnic3 DvsPortset-1         50.35    0.23       2.86    0.00   0.00   0.00
  50331655                 vmk1     vmnic3 DvsPortset-1          2.67    0.07       3.43    0.00   0.00   0.00
  50331656                 vmk2     vmnic3 DvsPortset-1          0.19    0.00       0.19    0.00   0.00   0.00
  50331657                 vmk3     vmnic2 DvsPortset-1          0.00    0.00       0.19    0.00   0.00   0.00
  50331658 36847:web-sv-01a.eth     vmnic2 DvsPortset-1         21.93    0.11      20.03    0.06   0.00   0.00
  50331659          vdr-vdrPort     vmnic2 DvsPortset-1         41.01    0.27       0.00    0.00   0.00   0.00
  50331660 37475:app-sv-01a.eth     vmnic3 DvsPortset-1         15.26    0.11      17.17    0.06   0.00   0.00
  50331661 37531:db-sv-01a.eth0     vmnic3 DvsPortset-1          3.81    0.04       7.63    0.00   0.00   0.00
```

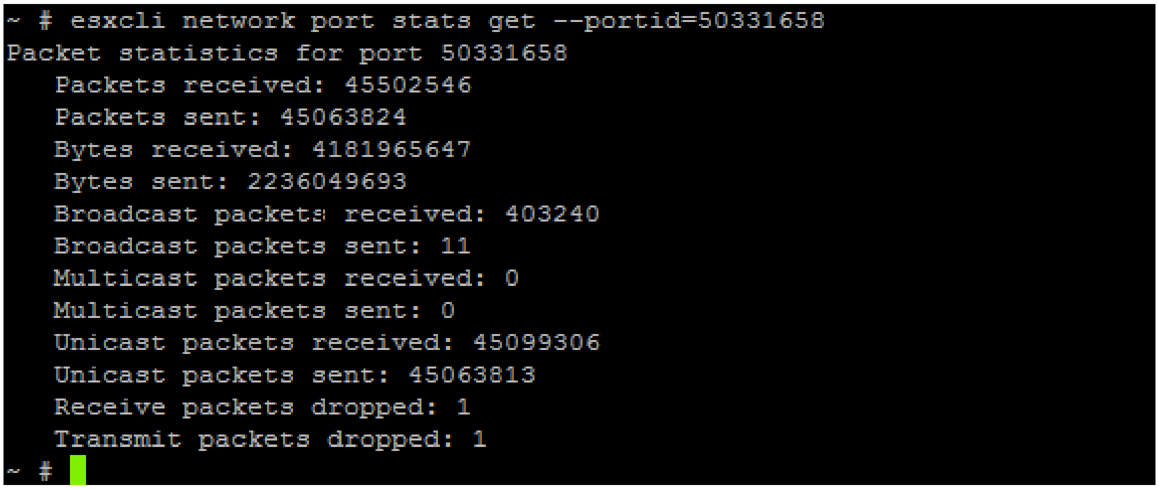
The **Port-ID** is a unique port number assigned to vNICs, vmknics or vmnics. **USED-BY** provides details of the VM, vmkernel NIC or vmnic that is connected to the vswitch port. The statistics are shown in the last six columns on the right.

In our example, let’s observe the statistics for VM - **web-sv-01a.** We see there are some packets Transmitted and received by this VM on its vNIC.

Traffic statistics such as packet counters for Unicast, Broadcast or Multicast traffic, can be obtained using the **esxcli network** command.

We will note the Port ID for web-sv-01a from the esxtop command and use it in our next command to see the packet counters.

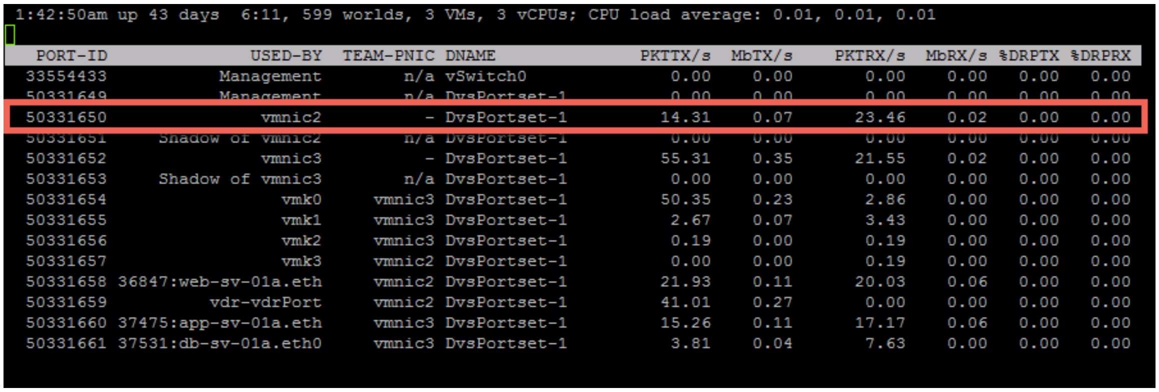
```
Syntax : esxcli network port stats get --portid=<PORT_ID>
```



```
~ # esxcli network port stats get --portid=50331658
Packet statistics for port 50331658
   Packets received: 45502546
   Packets sent: 45063824
   Bytes received: 4181965647
   Bytes sent: 2236049693
   Broadcast packets received: 403240
   Broadcast packets sent: 11
   Multicast packets received: 0
   Multicast packets sent: 0
   Unicast packets received: 45099306
   Unicast packets sent: 45063813
   Receive packets dropped: 1
   Transmit packets dropped: 1
~ #
```

To monitor statistics for Hypervisor Uplinks (vmnic), we can use the same set of commands as described in the above section.

esxtop + n will provide the packets/s, bytes/s, dropped packets % statistics for vmnics. The Port-ID can be noted for the vmnic that is to be monitored. In our example, lets monitor **vmnic2**.



```
 1:42:50am up 43 days  6:11, 599 worlds, 3 VMs, 3 vCPUs; CPU load average: 0.01, 0.01, 0.01

   PORT-ID              USED-BY  TEAM-PNIC DNAME              PKTTX/s  MbTX/s    PKTRX/s  MbRX/s %DRPTX %DRPRX
  33554433           Management        n/a vSwitch0              0.00    0.00       0.00    0.00   0.00   0.00
  50331649           Management        n/a DvsPortset-1          0.00    0.00       0.00    0.00   0.00   0.00
  50331650               vmnic2          - DvsPortset-1         14.31    0.07      23.46    0.02   0.00   0.00
  50331651     Shadow of vmnic2        n/a DvsPortset-1          0.00    0.00       0.00    0.00   0.00   0.00
  50331652               vmnic3          - DvsPortset-1         55.31    0.35      21.55    0.02   0.00   0.00
  50331653     Shadow of vmnic3        n/a DvsPortset-1          0.00    0.00       0.00    0.00   0.00   0.00
  50331654                 vmk0     vmnic3 DvsPortset-1         50.35    0.23       2.86    0.00   0.00   0.00
  50331655                 vmk1     vmnic3 DvsPortset-1          2.67    0.07       3.43    0.00   0.00   0.00
  50331656                 vmk2     vmnic3 DvsPortset-1          0.19    0.00       0.19    0.00   0.00   0.00
  50331657                 vmk3     vmnic2 DvsPortset-1          0.00    0.00       0.19    0.00   0.00   0.00
  50331658 36847:web-sv-01a.eth     vmnic2 DvsPortset-1         21.93    0.11      20.03    0.06   0.00   0.00
  50331659         vdr-vdrPort      vmnic2 DvsPortset-1         41.01    0.27       0.00    0.00   0.00   0.00
  50331660 37475:app-sv-01a.eth     vmnic3 DvsPortset-1         15.26    0.11      17.17    0.06   0.00   0.00
  50331661 37531:db-sv-01a.eth0     vmnic3 DvsPortset-1          3.81    0.04       7.63    0.00   0.00   0.00
```

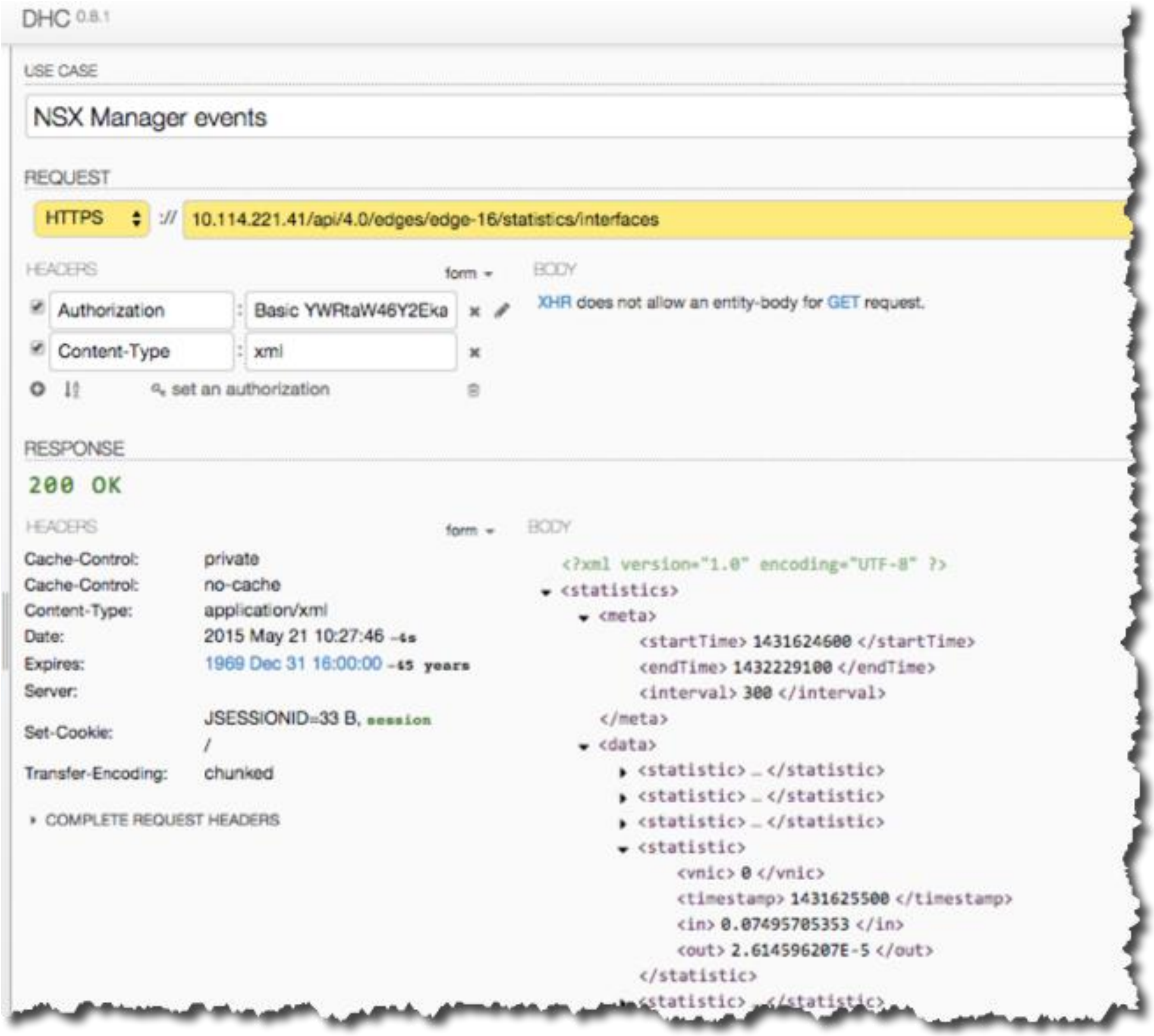
The Port-ID corresponding to vmnic2 is noted and used in our **esxcli network** command.

NSX API monitoring of interface statistics

The NSX REST API can be used for capturing statistics that monitor interfaces and other resources in NSX. Given below is an example of executing an API call to capture the interface statistics on an edge device with an ID *edge-16*. The API call for this looks like:

```
GET https://admin:admin@10.114.221.41/api/4.0/edges/edge-16/statistics/interfaces
```

The response to the API call has the statistics for all the interfaces. The figure below shows the statistics for *vnic0*, as displayed in the API response.



NSX Manager events

HTTPS :// 10.114.221.41/api/4.0/edges/edge-16/statistics/interfaces

Authorization	Basic YWRtaW46Y2Eka
Content-Type	xml

XHR does not allow an entity-body for GET request.

200 OK

```
Cache-Control:       private
Cache-Control:       no-cache
Content-Type:        application/xml
Date:                2015 May 21 10:27:46 -4s
Expires:             1969 Dec 31 16:00:00 -45 years
Server:
Set-Cookie:          JSESSIONID=33 B, session
                     /
Transfer-Encoding:   chunked
```

```
<?xml version="1.0" encoding="UTF-8" ?>
▾ <statistics>
  ▾ <meta>
      <startTime> 1431624600 </startTime>
      <endTime> 1432229100 </endTime>
      <interval> 300 </interval>
    </meta>
  ▾ <data>
    ▸ <statistic> … </statistic>
    ▸ <statistic> … </statistic>
    ▸ <statistic> … </statistic>
    ▾ <statistic>
        <vnic> 0 </vnic>
        <timestamp> 1431625500 </timestamp>
        <in> 0.07495705353 </in>
        <out> 2.614596207E-5 </out>
      </statistic>
    ▸ <statistic> … </statistic>
```

NSX APIs are available for collecting firewall statistics, tunnel statistics, and interface statistics. For a complete list of available APIs for statistics collection, please refer to the NSX-v API Guide.

vSphere API Monitoring of distributed interface statistics

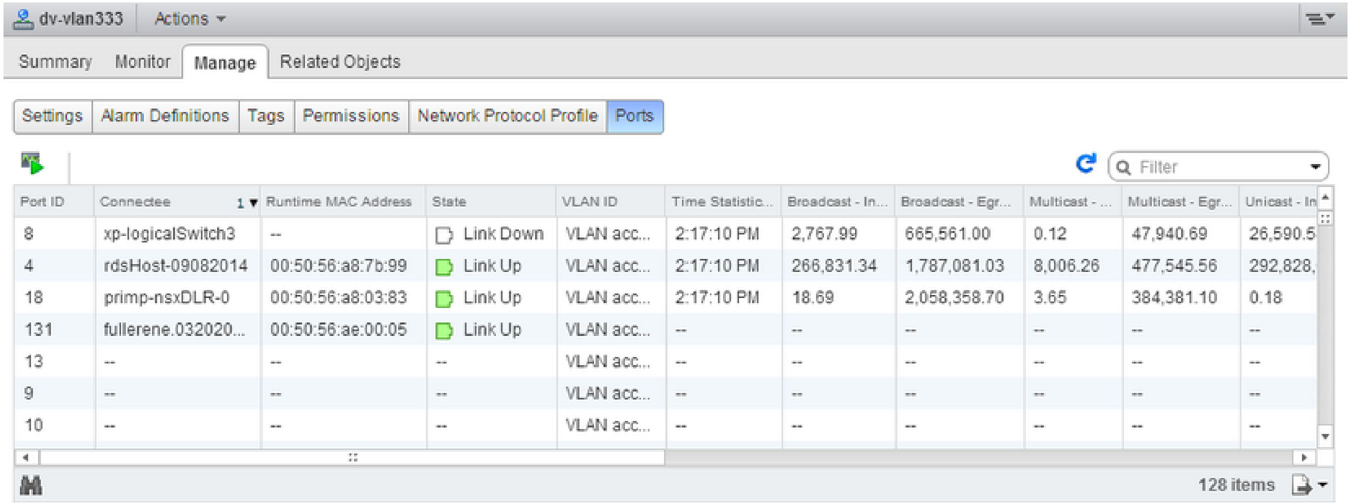
Along with the NSX REST API and NSX CLI, you can use the **vSphere API** to monitor certain NSX operations. The vSphere API is exposed as a web service, running on VMware vSphere server systems. The API provides access to the vSphere management components—the managed objects that you can use to manage, monitor, and control life-cycle operations of virtual machines and other virtual infrastructure components (datacenters, datastores, networks, and so on).

This vSphere API Reference is a core component of the VMware vSphere Web Services SDK. You can use the vSphere Web Services SDK to create custom solutions for managing virtual components, and to integrate existing datacenter management solutions with VMware technologies. For example, using the vSphere API, you can quickly create, customize, or migrate virtual machines.

Included below are a few examples of vSphere APIs and the scripts that can be used to periodically use the APIs for monitoring purposes.

View the statistics of an individual Distributed Virtual Port

To view the statistics of an individual Distributed Virtual Port (the data object is DistributedVirtualSwitchPortStatistics), you must drill down into a specific distributed virtual port group which you can find using either the vSphere Web client or C# Client. Below is a screenshot from a vSphere Web Client.



Port ID	Connectee	Runtime MAC Address	State	VLAN ID	Time Statistic...	Broadcast - In...	Broadcast - Egr...	Multicast - ...	Multicast - Egr...	Unicast - In
8	xp-logicalSwitch3	--	Link Down	VLAN acc...	2:17:10 PM	2,767.99	665,561.00	0.12	47,940.69	26,590.5
4	rdsHost-09082014	00:50:56:a8:7b:99	Link Up	VLAN acc...	2:17:10 PM	266,831.34	1,787,081.03	8,006.26	477,545.56	292,828,
18	primp-nsxDLR-0	00:50:56:a8:03:83	Link Up	VLAN acc...	2:17:10 PM	18.69	2,058,358.70	3.65	384,381.10	0.18
131	fullerene.032020...	00:50:56:ae:00:05	Link Up	VLAN acc...	--	--	--	--	--	--
13	--	--	--	VLAN acc...	--	--	--	--	--	--
9	--	--	--	VLAN acc...	--	--	--	--	--	--
10	--	--	--	VLAN acc...	--	--	--	--	--	--

To retrieve information on a Distributed Virtual Port using the vSphere API, you can refer to the DistributedVirtualPort property. Within this object, there is a state property that you can inspect to retrieve statistics using the stat property. If you would like to be able to pull other properties as shown in the screenshot, you can use the runtimeInfo property that provides additional information on the Distributed Virtual Port, such as connected entity, MAC Address, VLAN ID, state, and so on.

Below is a simple PowerCLI script which connects to a vCenter Server and retrieves a specific Distributed Virtual Portgroup (dvPg) that we are interested in using the Get-VDPortgroup cmdlet.

The output of this script capturing the statistics for distributed virtual port 4 is as below.

Get distributed virtual port statistics

The python script below uses the vSphere API to get the distributed virtual port statistics (the data object is DistributedVirtualSwitchPortStatistics) for all available distributed virtual ports. The output follows the script.

Script: "get_dvport_stats.py"

```
#-------------------------------------------------------------------------------
# Name:          get_dvport_stats
# Purpose:       Collect the statistics of all active dvPorts
#                and print it after every 10 seconds
#-------------------------------------------------------------------------------
```

```
from pyVmomi import vim, vmodl

from pyVim import connect

import atexit

from time import gmtime, strftime, sleep


class vcenter():

    def __init__(self, vcenter_host, vcenter_username, vcenter_passwd):

        self.vc_host = vcenter_host

        self.vc_username = vcenter_username

        self.vc_passwd = vcenter_passwd


    # Function to connect to Vcenter

    def connect_vcenter(self):

        try:

            si = connect.Connect(self.vc_host, 443, self.vc_username , self.vc_passwd,
service="hostd")

            # Disconnect connection while existing from program

            atexit.register(connect.Disconnect, si)

        # Catch exception if username or password is incorrect

        except vim.fault.InvalidLogin as e:

            print e.msg

            return -1

        # Catch exception if Vcenter host name is invalid

        except vim.fault.HostConnectFault as e:

            print 'Invalid Host'

            return -1

        else:

            # Return the object of connection

            return si


    # Function to get statistics of Distributed Virtual Port

    def get_dvportstats(self, si):

        # Retrive contents of Vcenter

        content = si.RetrieveContent()
```

```
list_portstats = []
# Get the objects of DVS
vs_view = content.viewManager.CreateContainerView(content.rootFolder,
                                                [vim.DistributedVirtualSwitch], True)
dvs_objects = vs_view.view
# If no dvs present in Vcenter
if not dvs_objects:
    print 'No DVS available!'
    exit(0)
else:
    print 'Virtual Port Statistics::'
    print '----------------------------------------------------------'
    for dvs in dvs_objects:
        dict_portstats = {}
        # get objects of Distributed virtual port
        ports = dvs.FetchDVPorts()
        for port in ports:
            # Get the state of each port
            port_state = port.state
            if port_state is None:
                continue
            else:
                # Get Stats and runtimeInfo of port
                # Stats and runtimeInfo are properties of port state
                port_stats = port_state.stats
                runtime_info = port_state.runtimeInfo
                #Create a dictionary of Stats and runtimeInfo
                if port_stats and runtime_info:
                    dict_portstats = {'runtime_info' : runtime_info,
                                      'stats' : port_stats}
                    # Append the dictionary in port list
                    list_portstats.append(dict_portstats)
# Return the list
return list_portstats
```

```
def print_stats(list_portstats):
    #Print statistics of dvPort
    for port in list_portstats:
        print '\n\tLink Up :', port["runtime_info"].linkUp
        print '\tRun time MAC address :', port["runtime_info"].macAddress
        print '\n\tBytesInBroadcast :', port["stats"].bytesInBroadcast
        print '\tBytesInMulticast :', port["stats"].bytesInMulticast
        print '\tBytesInUnicast :', port["stats"].bytesInUnicast
        print '\tBytesOutBroadcast :', port["stats"].bytesOutBroadcast
        print '\tBytesOutMulticast :', port["stats"].bytesOutMulticast
        print '\tBytesOutUnicast :', port["stats"].bytesOutUnicast
        print '\tPacketsInBroadcast :', port["stats"].packetsInBroadcast
        print '\tPacketsInDropped :', port["stats"].packetsInDropped
        print '\tPacketsInException :', port["stats"].packetsInException
        print '\tPacketsInMulticast :', port["stats"].packetsInMulticast
        print '\tPacketsInUnicast :', port["stats"].packetsInUnicast
        print '\tPacketsOutBroadcast :', port["stats"].packetsOutBroadcast
        print '\tPacketsOutDropped :', port["stats"].packetsOutDropped
        print '\tPacketsOutException :', port["stats"].packetsOutException
        print '\tPacketsOutMulticast :', port["stats"].packetsOutMulticast
        print '\tPacketsOutUnicast :', port["stats"].packetsOutUnicast
        print '----------------------------------------------------------'


def main ():
    """
    Creates object of vcenter class and call different methods
    of class.
    """

    # Host name, username and password of Vcenter are hardcoded here
    # We can take these inputs from user also
    vcenter_host = 'vc-l-01a.corp.local'
    vcenter_username = 'root'
    vcenter_passwd = 'VMware1!'
    # Create vcenter object by passing VC host name,username and password as arguments
    vc = vcenter(vcenter_host, vcenter_username, vcenter_passwd)
    # Function call to connect to Vcenter
    si = vc.connect_vcenter()
    # If -1 is returned, come out of program
    if si is -1 :
        exit(0)
    # Call get_dvportstats fuction after every 10 seconds

    while True :
        print '\nTimestamp :', strftime("%Y-%m-%d %H:%M:%S", gmtime())
        # Call get_dvportstats function which returns list of dictionary of statistics of
        # all active dvPorts in Vcenter
        list_portstats = vc.get_dvportstats(si)
        # Call print_stats to print statistics of all active ports
        print_stats(list_portstats)
        # Sleep for 10 seconds
        sleep(10)


if __name__ == '__main__':
    # Call main function
    main()
```

Output from script "get_dvport_stats.py"

Below, we show an abbreviated version of the output from the above port statistics-gathering script on an example installation:

```
Timestamp : 2015-07-13 13:28:28
Virtual Port Statistics::
----------------------------------------------------------

      Link Up : True
      Run time MAC address : 00:00:00:00:00:00

      BytesInBroadcast : 0
      BytesInMulticast : 0
      BytesInUnicast : 4310
      BytesOutBroadcast : 0
      BytesOutMulticast : 0
      BytesOutUnicast : 0
      PacketsInBroadcast : 0
      PacketsInDropped : 0
      PacketsInException : 0
      PacketsInMulticast : 0
      PacketsInUnicast : 13
      PacketsOutBroadcast : 0
      PacketsOutDropped : 0
      PacketsOutException : 0
      PacketsOutMulticast : 0
      PacketsOutUnicast : 0
----------------------------------------------------------


      Link Up : True
      Run time MAC address : 00:00:00:00:00:00

      BytesInBroadcast : 0
      BytesInMulticast : 0
      BytesInUnicast : 73562
      BytesOutBroadcast : 0
      BytesOutMulticast : 0
      BytesOutUnicast : 0
      PacketsInBroadcast : 0
      PacketsInDropped : 0
      PacketsInException : 0
      PacketsInMulticast : 0
      PacketsInUnicast : 105
      PacketsOutBroadcast : 0
      PacketsOutDropped : 0
      PacketsOutException : 0
      PacketsOutMulticast : 0
      PacketsOutUnicast : 0
-----------------------------------------------------------
      Link Up : True
      Run time MAC address : 00:50:56:a6:1b:52

      BytesInBroadcast : 0
      BytesInMulticast : 0
      BytesInUnicast : 0
      BytesOutBroadcast : 0
      BytesOutMulticast : 0
      BytesOutUnicast : 0
      PacketsInBroadcast : 0
      PacketsInDropped : 0
      PacketsInException : 0
      PacketsInMulticast : 0
      PacketsInUnicast : 0
      PacketsOutBroadcast : 0
      PacketsOutDropped : 0
      PacketsOutException : 0
      PacketsOutMulticast : 0
      PacketsOutUnicast : 0
-----------------------------------------------------------
...
```

Show performance counters for vCenter

The python script below uses the vSphere API to provide performance counters for vCenter.

Data Object - PerfCounterInfo1

Sample script: perf_counter.py

```
#-------------------------------------------------------------------------------
# Name:        perf_counter.py
# Purpose:     Get the performance counters of Vcenter
#-------------------------------------------------------------------------------
```

```
from pyVmomi import vim, vmodl

from pyVim import connect

from time import gmtime, strftime, sleep

import atexit


class vcenter():

    def __init__(self, vcenter_host, vcenter_username, vcenter_passwd):

        self.vc_host = vcenter_host

        self.vc_username = vcenter_username

        self.vc_passwd = vcenter_passwd


    # Function to connect to Vcenter

    def connect_vcenter(self):

        try:

            si = connect.Connect(self.vc_host, 443, self.vc_username , self.vc_passwd,
service="hostd")

            # Disconnect connection while existing from program

            atexit.register(connect.Disconnect, si)

        # Catch exception if username or password is incorrect

        except vim.fault.InvalidLogin as e:

            print e.msg

            return -1

        # Catch exception if Vcenter host name is invalid

        except vim.fault.HostConnectFault as e:

            print 'Invalid Host'

            return -1

        else:

            # Return the object of connection

            return si


    # Function to get statistics of Distributed Virtual Port

    def get_perfcounter(self, si):

        # Retrive contents of VC
        content = si.RetrieveContent()
        # Create object of perfManager
        perf_mgr = content.perfManager
        # Get the list of perf counters
        perf_counter = perf_mgr.perfCounter
        return perf_counter


def print_perfcounter(perf_counter):
        # Print various perf counters
        for counter in perf_counter:
            name_info = counter.nameInfo
            group_info = counter.groupInfo
            unit_info = counter.unitInfo
            print '\tCounterId :', counter.key
            print '\tName Info :'
            print '\t\tLable :', name_info.label
            print '\t\tSummary :', name_info.summary
            print '\t\tKey :',name_info.key
            print '\tGroup Info :'
            print '\t\tLable :', group_info.label
            print '\t\tSummary :',group_info.summary
            print '\t\tKey :',group_info.key
            print '\tUnit Info :'
            print '\t\tLable :', unit_info.label
            print '\t\tSummary :',unit_info.summary
            print '\t\tKey :',unit_info.key
            print '\trollupType :', counter.rollupType
            print '\tstatsType :', counter.statsType
            print '\tlevel :', counter.level
            print '\tperDeviceLevel :', counter.perDeviceLevel
            print '\tassociatedCounterId :', counter.associatedCounterId
            print '-----------------------------------------------------'
```

```
def main():
    """
    Creates object of vcenter class and call different methods
    of class.
    """

    # Host name, username and password of Vcenter are hardcoded here
    # We can take these inputs from user also
    vcenter_host = 'vc-l-01a.corp.local'
    vcenter_username = 'root'
    vcenter_passwd = 'VMware1!'
    # Create vcenter object by passing VC host name,username and password as arguments
    vc = vcenter(vcenter_host, vcenter_username, vcenter_passwd)
    # Function call to connect to Vcenter
    si = vc.connect_vcenter()
    # If -1 is returned, come out of program
    if si is -1 :
        exit(0)
    while True :
        # Print timestamp at which the stats at=re collected
        print '\nTimestamp :', strftime("%Y-%m-%d %H:%M:%S", gmtime())
        print'\nPerformance Counters ::: '
        print '-----------------------------------------------------'
        perf_counter = vc.get_perfcounter(si)
        print_perfcounter(perf_counter)
        sleep(10)


if __name__ == '__main__':
    # Call main function
    main()
```

Sample output from perf_counter.py

```
Performance Counters :::

...

-----------------------------------------------------

      CounterId : 142

      Name Info :

             Lable : Usage

             Summary : Network utilization (combined transmit-rates and receive-rates)
during the interval

             Key : usage

      Group Info :

             Lable : Network

             Summary : Network

             Key : net

      Unit Info :

             Lable : KBps

             Summary : Kilobytes per second

             Key : kiloBytesPerSecond

      rollupType : none

      statsType : rate

      level : 4

      perDeviceLevel : 4

      associatedCounterId : (int) []

-----------------------------------------------------

...

-----------------------------------------------------

      CounterId : 146

      Name Info :

             Lable : Packets received

             Summary : Number of packets received during the interval

             Key : packetsRx

      Group Info :

             Lable : Network

             Summary : Network
```

```
              Key : net
       Unit Info :
              Lable : Number
              Summary : Number
              Key : number
       rollupType : summation
       statsType : delta
       level : 2
       perDeviceLevel : 3
       associatedCounterId : (int) []
-----------------------------------------------------
       CounterId : 147
       Name Info :
              Lable : Packets transmitted
              Summary : Number of packets transmitted during the interval
              Key : packetsTx
       Group Info :
              Lable : Network
              Summary : Network
              Key : net
       Unit Info :
              Lable : Number
              Summary : Number
              Key : number
       rollupType : summation
       statsType : delta
       level : 2
       perDeviceLevel : 3
       associatedCounterId : (int) []
-----------------------------------------------------
       CounterId : 148
       Name Info :
              Lable : Data receive rate
              Summary : Average rate at which data was received during the interval
              Key : received
       Group Info :
              Lable : Network
              Summary : Network
              Key : net
       Unit Info :
              Lable : KBps
              Summary : Kilobytes per second
              Key : kiloBytesPerSecond
       rollupType : average
       statsType : rate
       level : 2
       perDeviceLevel : 3
       associatedCounterId : (int) []
-----------------------------------------------------
       CounterId : 149
       Name Info :
              Lable : Data transmit rate
              Summary : Average rate at which data was transmitted during the interval
              Key : transmitted
       Group Info :
              Lable : Network
              Summary : Network
              Key : net
       Unit Info :
              Lable : KBps
              Summary : Kilobytes per second
              Key : kiloBytesPerSecond
       rollupType : average
       statsType : rate
       level : 2
       perDeviceLevel : 3
       associatedCounterId : (int) []
-----------------------------------------------------
       CounterId : 150
       Name Info :
              Lable : pNic Throughput Provisioned
              Summary : Provisioned pNic I/O Throughput
              Key : throughput.provisioned
       Group Info :
              Lable : Network
              Summary : Network
              Key : net
       Unit Info :
              Lable : KBps
              Summary : Kilobytes per second
              Key : kiloBytesPerSecond
       rollupType : average
       statsType : absolute
       level : 4
       perDeviceLevel : 4
       associatedCounterId : (int) []
----------------------------------------------------
       CounterId : 151
       Name Info :
              Lable : pNic Throughput Usable
              Summary : Usable pNic I/O Throughput
              Key : throughput.usable
       Group Info :
              Lable : Network
              Summary : Network
              Key : net
       Unit Info :
              Lable : KBps
              Summary : Kilobytes per second
              Key : kiloBytesPerSecond
       rollupType : average
       statsType : absolute
       level : 4
       perDeviceLevel : 4
...
```

Capture virtual machine statistics

The following script captures virtual machine statistics using vSphere API.

Data Object – VirtualMachineQuickStats

Sample script: vm_quickstats.py

```
#-------------------------------------------------------------------------------

# Name:         vm_quickstats.py

# Purpose:      Collect the statistics of all VMs in VCenter

#               and print it after every 10 seconds

#-------------------------------------------------------------------------------


from pyVmomi import vim, vmodl

from pyVim import connect

import atexit

from time import gmtime, strftime, sleep


class vcenter():

    def __init__(self, vcenter_host, vcenter_username, vcenter_passwd):

        self.vc_host = vcenter_host

        self.vc_username = vcenter_username

        self.vc_passwd = vcenter_passwd


    # Function to connect to Vcenter

    def connect_vcenter(self):

        try:

            si = connect.Connect(self.vc_host, 443, self.vc_username , self.vc_passwd, 
service="hostd")

            # Disconnect connection while existing from program

            atexit.register(connect.Disconnect, si)

        # Catch exception if username or password is incorrect

        except vim.fault.InvalidLogin as e:

            print e.msg

            return -1

        # Catch exception if Vcenter host name is invalid

        except vim.fault.HostConnectFault as e:
```

```
        print 'Invalid Host'
        return -1
    else:
        # Return the object of connection
        return si

# Function to get statistics of all VMs
def vm_quickstat(self, si):
    vm_list = []
    # Retrive contents of Vcenter
    content = si.RetrieveContent()
    # Get the objects of VMs
    vs_view = content.viewManager.CreateContainerView(content.rootFolder,
                                                     [vim.VirtualMachine], True)
    vm_objects = vs_view.view
    # If no VM present in Vcenter
    if not vm_objects:
        print 'No vm available!'
        exit(0)
    else:
        for vm in vm_objects:
            dict_vm = {}
            # Get the name of VM
            vm_name = vm.name
            # Get summary of VM
            vm_summary = vm.summary
            # Get quickstats of each VM
            # quickStats is a property of summary
            vm_quickstats = vm_summary.quickStats
            # Create a dictionary of vm_name and quickstats
            dict_vm = {'vm_name' : vm_name,
                       'stats' : vm_quickstats}
            # Append dictionary for each VM in vm_list
            vm_list.append(dict_vm)
    return vm_list
```

```
def print_stats(vm_list):
    # Print name and statistics of VM
    for vm in vm_list:
        print '\n\tVM Name :', vm["vm_name"]
        print '\n\tBalloonedMemory :', vm["stats"].balloonedMemory
        print '\tCompressedMemory :', vm["stats"].compressedMemory
        print '\tConsumedOverheadMemory :', vm["stats"].consumedOverheadMemory
        print '\tDistributedCpuEntitlement :', vm["stats"].distributedCpuEntitlement
        print '\tDistributedMemoryEntitlement :', vm["stats"].distributedMemoryEntitlement
        print '\tftLatencyStatus :', vm["stats"].ftLatencyStatus
        print '\tftLogBandwidth :', vm["stats"].ftLogBandwidth
        print '\tftSecondaryLatency :', vm["stats"].ftSecondaryLatency
        print '\tGuestHeartbeatStatus :', vm["stats"].guestHeartbeatStatus
        print '\tGuestMemoryUsage :', vm["stats"].guestMemoryUsage
        print '\tHostMemoryUsage :', vm["stats"].hostMemoryUsage
        print '\tOverallCpuDemand :', vm["stats"].overallCpuDemand
        print '\tOverallCpuUsage :', vm["stats"].overallCpuUsage
        print '\tPrivateMemory :', vm["stats"].privateMemory
        print '\tSharedMemory :', vm["stats"].sharedMemory
        print '\tssdSwappedMemory :', vm["stats"].ssdSwappedMemory
        print '\tStaticCpuEntitlement :', vm["stats"].staticCpuEntitlement
        print '\tStaticMemoryEntitlement :', vm["stats"].staticMemoryEntitlement
        print '\tSwappedMemory :', vm["stats"].swappedMemory
        print '\tUptimeSeconds :', vm["stats"].uptimeSeconds
        print '---------------------------------------------------------------'


def main():
    """
    Creates object of vcenter class and call different methods
    of class.
    """
    # Host name, username and password of Vcenter are hardcoded here

    # We can take these inputs from user also

    vcenter_host = 'vc-l-01a.corp.local'

    vcenter_username = 'root'

    vcenter_passwd = 'VMware1!'

    # Create vcenter object by passing VC host name,username and password as arguments

    vc = vcenter(vcenter_host, vcenter_username, vcenter_passwd)

    # Function call to connect to Vcenter

    si = vc.connect_vcenter()

    # If -1 is returned, come out of program

    if si is -1 :

        exit(0)



    while True :

        # Print timestamp at which the stats at=re collected

        print '\nTimestamp :', strftime("%Y-%m-%d %H:%M:%S", gmtime())

        print 'Virtual Machine Statistics ::'

        # Call vm_quickstat function to collect statistics of all VMs

        vm_list = vc.vm_quickstat(si)

        # Call print_stats to print statistics of all VMs

        print_stats(vm_list)

        sleep(10)



if __name__ == '__main__':

    # Call main function

    main()
```

Output from sample script, vm_quickstats.py

```
Virtual Machine Statistics ::


      VM Name : web-sv-02a
```

```
       BalloonedMemory : 0
       CompressedMemory : 0
       ConsumedOverheadMemory : 22
       DistributedCpuEntitlement : 0
       DistributedMemoryEntitlement : 313
       ftLatencyStatus : gray
       ftLogBandwidth : -1
       ftSecondaryLatency : -1
       GuestHeartbeatStatus : green
       GuestMemoryUsage : 20
       HostMemoryUsage : 194
       OverallCpuDemand : 0
       OverallCpuUsage : 0
       PrivateMemory : 172
       SharedMemory : 0
       ssdSwappedMemory : None
       StaticCpuEntitlement : 1256
       StaticMemoryEntitlement : 339
       SwappedMemory : 0
       UptimeSeconds : 1566167
---------------------------------------------------------------


       VM Name : av-win7-01a


       BalloonedMemory : 0
       CompressedMemory : 0
       ConsumedOverheadMemory : 31
       DistributedCpuEntitlement : 26
       DistributedMemoryEntitlement : 585
       ftLatencyStatus : gray
       ftLogBandwidth : -1
       ftSecondaryLatency : -1
       GuestHeartbeatStatus : green
       GuestMemoryUsage : 51
       HostMemoryUsage : 1055

       OverallCpuDemand : 26

       OverallCpuUsage : 26

       PrivateMemory : 1024

       SharedMemory : 0

       ssdSwappedMemory : None

       StaticCpuEntitlement : 1252

       StaticMemoryEntitlement : 679

       SwappedMemory : 0

       UptimeSeconds : 1566181
---------------------------------------------------------------

...
```

For a list of vSphere APIs and all the comprehensive information about all data structures available through the vSphere APIs, please refer to the VMware vSphere API Reference Guide.

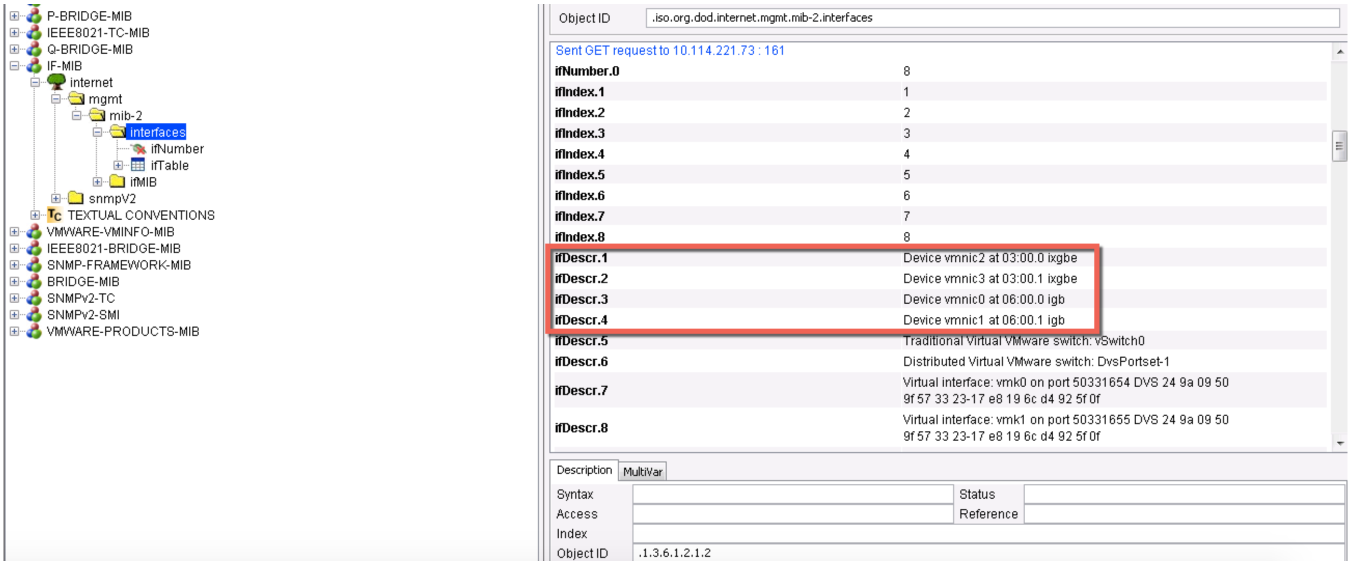
Hypervisor Interface Statistics

SNMP

Hypervisor Interfaces (vmnic) status and statistics can be monitored using standard IF-MIB.

Object .iso.org.dod.internet.mgmt.mib-2.interfaces provides details of the interfaces such as interface index (ifIndex) and description (ifDescr). Identify the vmnic interfaces that of interest and note the corresponding index numbers.

In our example, we have two 10G NICs (vmnic2 and vmnic3) which are connected to the Top-Of-Rack switch. The ifIndex values for these two vmnics are **1** and **2.**

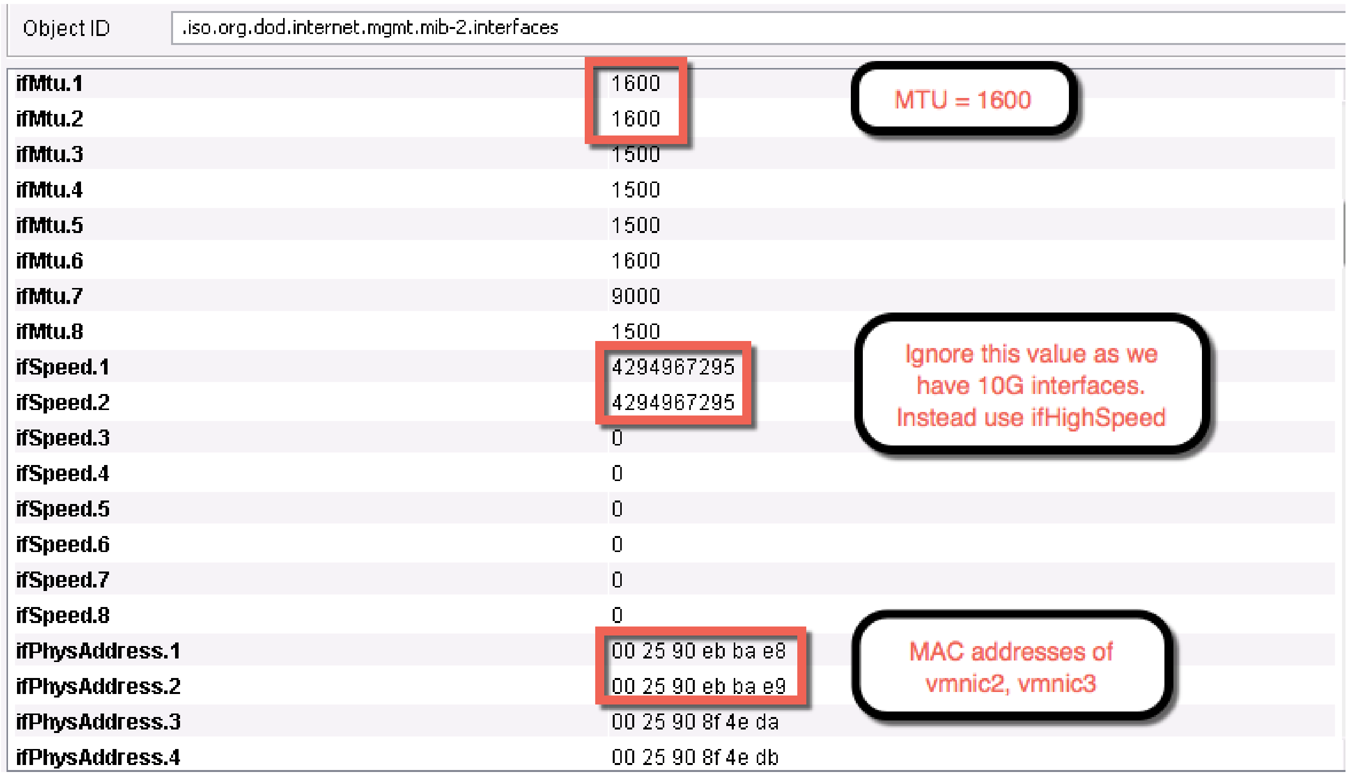


```
P-BRIDGE-MIB
IEEE8021-TC-MIB
Q-BRIDGE-MIB
IF-MIB
  internet
    mgmt
      mib-2
        interfaces
          ifNumber
          ifTable
        ifMIB
    snmpV2
  TEXTUAL CONVENTIONS
VMWARE-VMINFO-MIB
IEEE8021-BRIDGE-MIB
SNMP-FRAMEWORK-MIB
BRIDGE-MIB
SNMPv2-TC
SNMPv2-SMI
VMWARE-PRODUCTS-MIB
```

```
Object ID   .iso.org.dod.internet.mgmt.mib-2.interfaces

Sent GET request to 10.114.221.73 : 161
ifNumber.0      8
ifIndex.1       1
ifIndex.2       2
ifIndex.3       3
ifIndex.4       4
ifIndex.5       5
ifIndex.6       6
ifIndex.7       7
ifIndex.8       8
ifDescr.1       Device vmnic2 at 03:00.0 ixgbe
ifDescr.2       Device vmnic3 at 03:00.1 ixgbe
ifDescr.3       Device vmnic0 at 06:00.0 igb
ifDescr.4       Device vmnic1 at 06:00.1 igb
ifDescr.5       Traditional Virtual VMware switch: vSwitch0
ifDescr.6       Distributed Virtual VMware switch: DvsPortset-1
ifDescr.7       Virtual interface: vmk0 on port 50331654 DVS 24 9a 09 50
                9f 57 33 23-17 e8 19 6c d4 92 5f 0f
ifDescr.8       Virtual interface: vmk1 on port 50331655 DVS 24 9a 09 50
                9f 57 33 23-17 e8 19 6c d4 92 5f 0f

Description | MultiVar
Syntax                    Status
Access                    Reference
Index
Object ID   .1.3.6.1.2.1.2
```

This object also provides details such as MTU, Speed and MAC Address as shown in figure below. In our case the MTU is configure for 1600 bytes. The ifSpeed object reports the max value of 4,294,967,295 since the NIC speed is more than 1000Mb/s (10Gbps in our setup). In such a case we should poll ifHighSpeed object to report the correct speed of the interfaces.

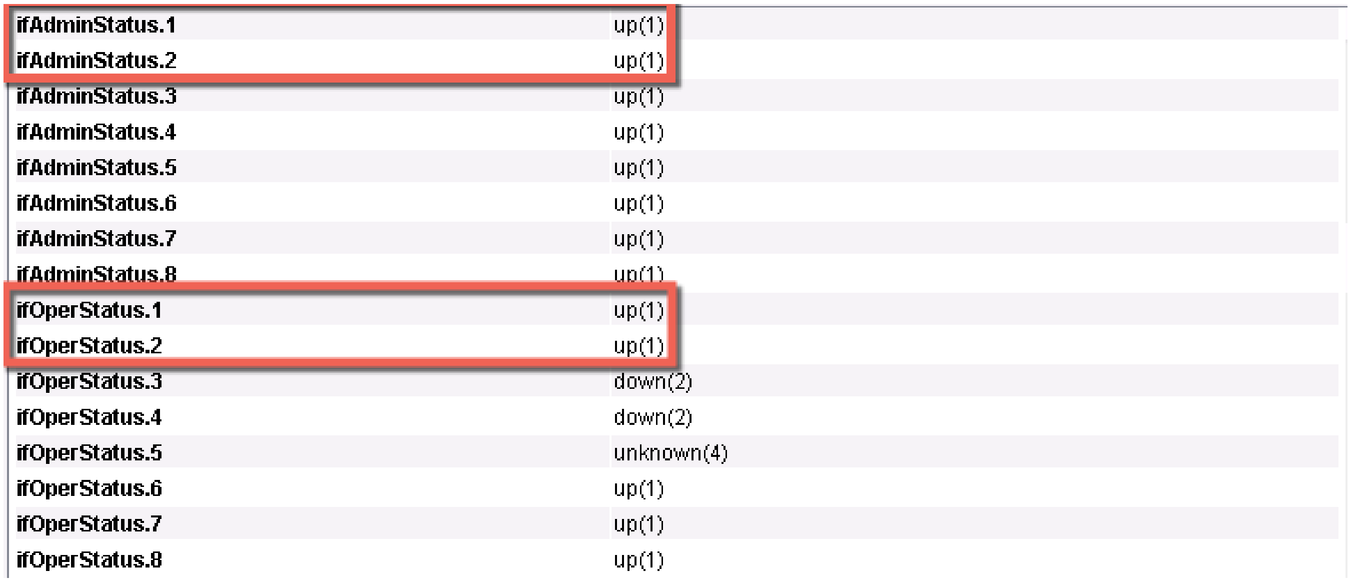
The MAC addresses of vmnic2 and vmnic3 are also reported in the figure below.



```
Object ID   .iso.org.dod.internet.mgmt.mib-2.interfaces

ifMtu.1             1600
ifMtu.2             1600
ifMtu.3             1500
ifMtu.4             1500
ifMtu.5             1500
ifMtu.6             1600
ifMtu.7             9000
ifMtu.8             1500
ifSpeed.1           4294967295
ifSpeed.2           4294967295
ifSpeed.3           0
ifSpeed.4           0
ifSpeed.5           0
ifSpeed.6           0
ifSpeed.7           0
ifSpeed.8           0
ifPhysAddress.1     00 25 90 eb ba e8
ifPhysAddress.2     00 25 90 eb ba e9
ifPhysAddress.3     00 25 90 8f 4e da
ifPhysAddress.4     00 25 90 8f 4e db
```

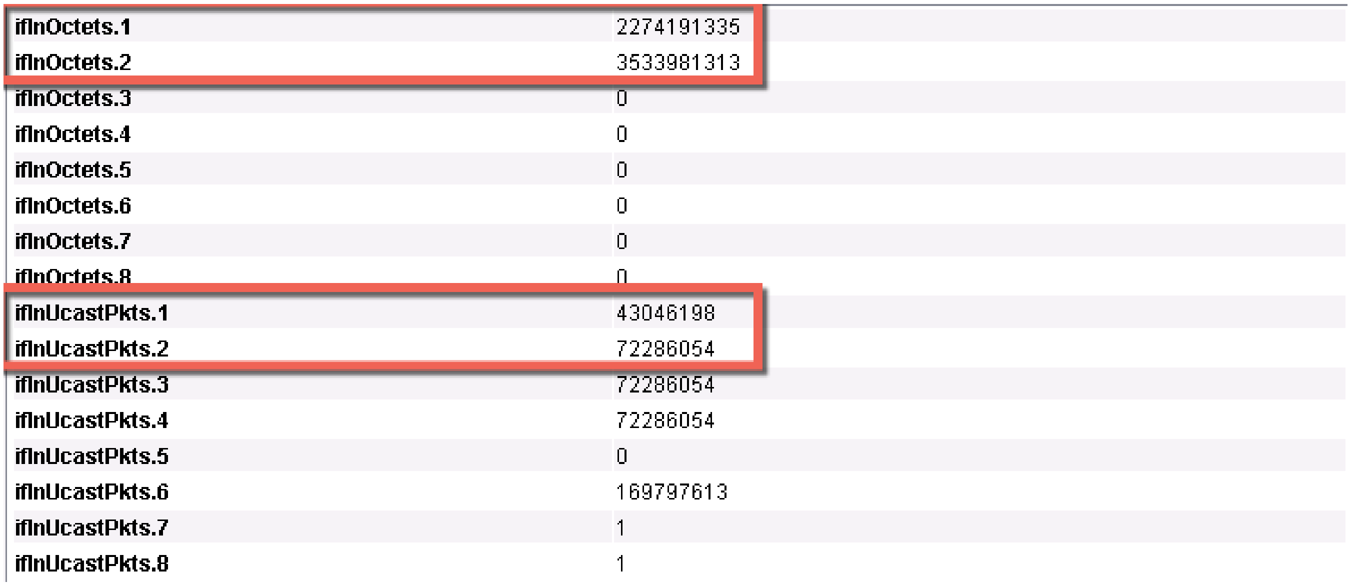
MTU = 1600
Ignore this value as we have 10G interfaces. Instead use ifHighSpeed
MAC addresses of vmnic2, vmnic3

Administrative and Operational Status for vmnic2 and vmnic3 is reported as UP as shown in figure below.



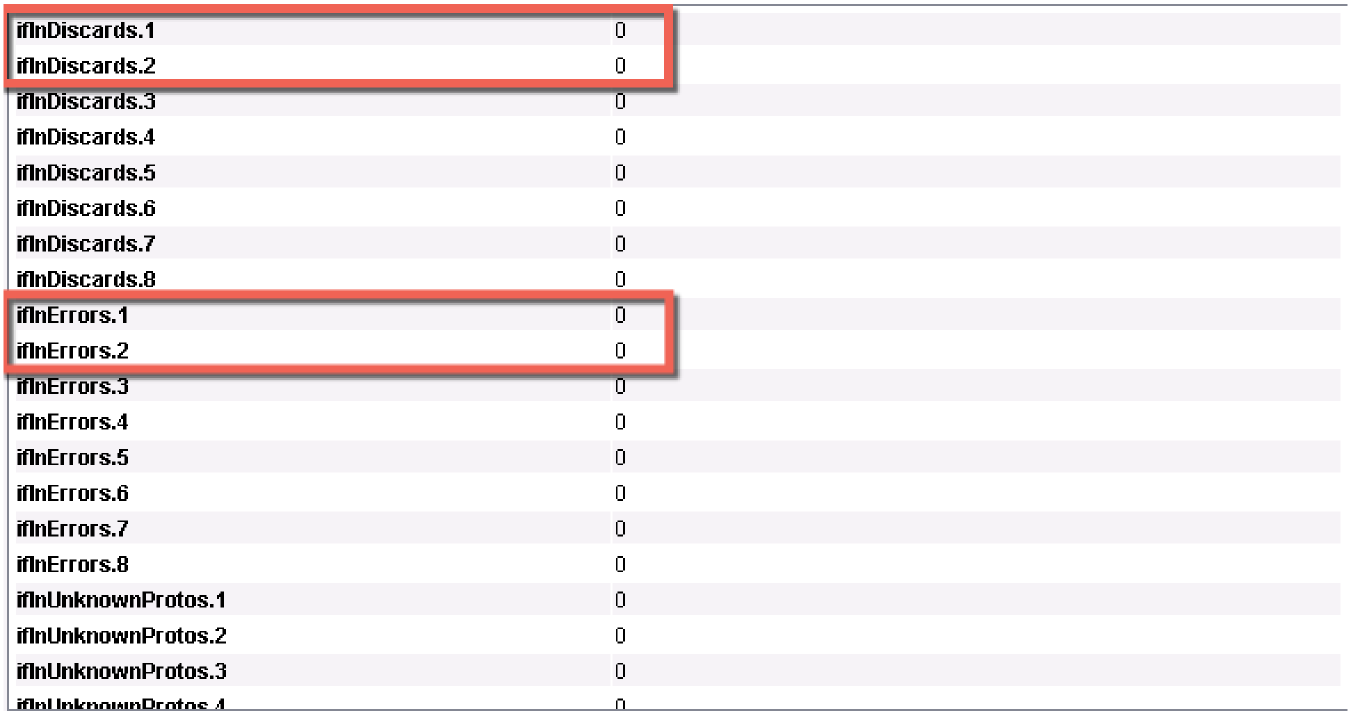
```
ifAdminStatus.1          up(1)
ifAdminStatus.2          up(1)
ifAdminStatus.3          up(1)
ifAdminStatus.4          up(1)
ifAdminStatus.5          up(1)
ifAdminStatus.6          up(1)
ifAdminStatus.7          up(1)
ifAdminStatus.8          up(1)
ifOperStatus.1           up(1)
ifOperStatus.2           up(1)
ifOperStatus.3           down(2)
ifOperStatus.4           down(2)
ifOperStatus.5           unknown(4)
ifOperStatus.6           up(1)
ifOperStatus.7           up(1)
ifOperStatus.8           up(1)
```

Interface utilization in terms of octets can be monitored using **ifInOctets** and **ifOutOctets** objects. Interface Packet counters for unicast traffic will be reported by **ifInUcastPkts** and **ifOutUcastPkts** object. For packet counters for multicast and broadcast traffic, use the objects in **ifXTable**.



```
ifInOctets.1             2274191335
ifInOctets.2             3533981313
ifInOctets.3             0
ifInOctets.4             0
ifInOctets.5             0
ifInOctets.6             0
ifInOctets.7             0
ifInOctets.8             0
ifInUcastPkts.1          43046198
ifInUcastPkts.2          72286054
ifInUcastPkts.3          72286054
ifInUcastPkts.4          72286054
ifInUcastPkts.5          0
ifInUcastPkts.6          169797613
ifInUcastPkts.7          1
ifInUcastPkts.8          1
```

ifInDiscards/ifOutDiscards and ifInErrors/ifOutErrors objects will represent issues such as lack of buffer space or CRC errors.



ifInDiscards.1	0
ifInDiscards.2	0
ifInDiscards.3	0
ifInDiscards.4	0
ifInDiscards.5	0
ifInDiscards.6	0
ifInDiscards.7	0
ifInDiscards.8	0
ifInErrors.1	0
ifInErrors.2	0
ifInErrors.3	0
ifInErrors.4	0
ifInErrors.5	0
ifInErrors.6	0
ifInErrors.7	0
ifInErrors.8	0
ifInUnknownProtos.1	0
ifInUnknownProtos.2	0
ifInUnknownProtos.3	0
ifInUnknownProtos.4	0

To monitor uplink utilization for Broadcast or Multicast traffic, use the **IF-MIB::ifXTable** object as shown below. Objects **ifInMulticastPkts** (or ifHCInMulticastPkts) and **ifInBroadcastPkts** (or **ifHCInBroadcastPkts**) will provide packet counters for Incoming Multicast and Broadcast packets, respectively. (Note: **ifHCInMulticastPkts/ifHCInBroadcastPkts** are 64 bit version of the counters and may be used all times).

Flow Visibility

Application and Network performance monitoring tools utilize flow data and create dashboards that help level 1 support to get notified of any performance issue. Since the granularity of flow information is greater than statistics, this data helps narrowing down issue to a particular application or network.

These tools also help in capacity planning exercises where the historical usage is monitored in the customer environment and a report is generated to show which application is utilizing how much BW. This helps administrator to allocate or prioritize traffic for the various applications in the data center

NSX Supports IPFIX (Netflow 10) to export IP flow information to a collector. IPFIX can be enabled on VMware Distributed Switch (VDS) and Distributed Firewall (DFW).

IPFIX on Logical Switch

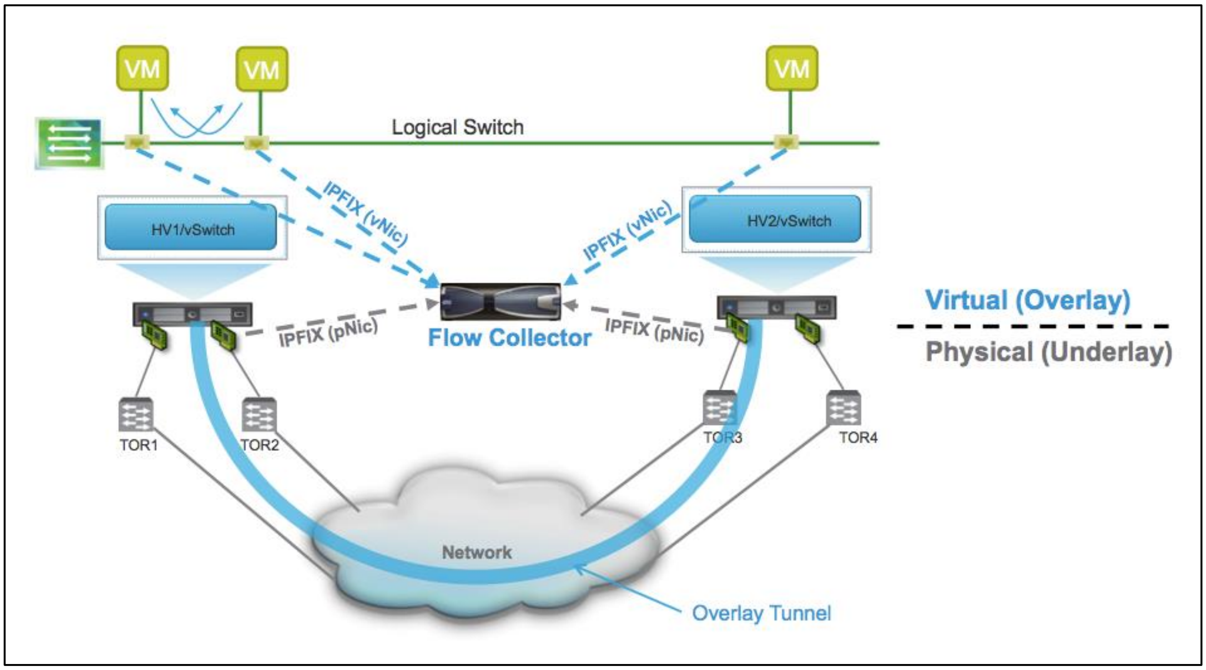


Figure: IPFIX from VDS

Enabling IPFIX for a NSX Logical Switch is a two-step process. The first step is to configure Netflow Collector on the VDS backing the NSX Transport zone (Logical Switch). The second step is to enable Netflow Monitoring on the dvPortGroup corresponding to the Logical Switch.

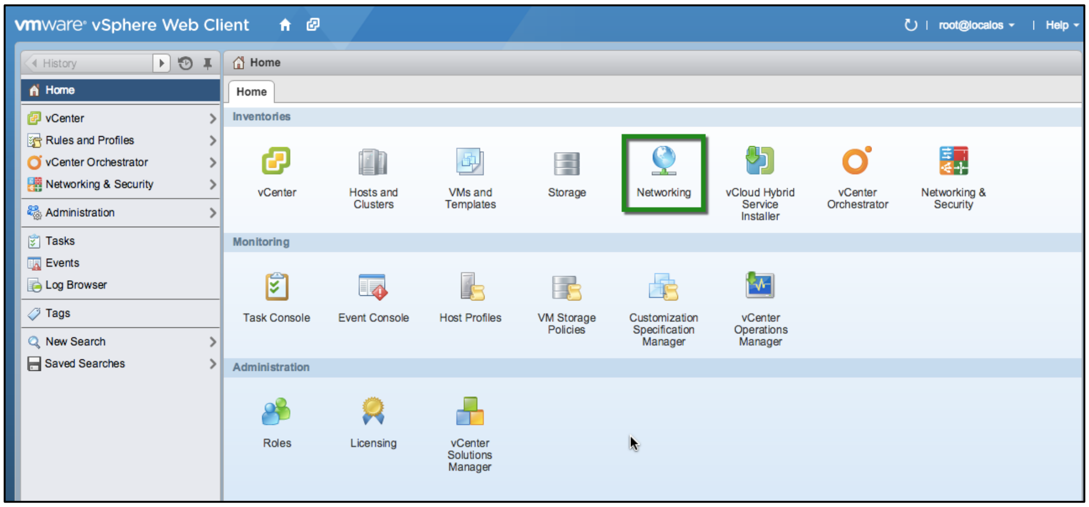
Note: If the NSX Transport Zone spans multiple VDS, then repeat these steps for each VDS/dvPortGroup.

In an NSX environment, the virtual machine data traffic on a Logical switch traversing the uplink of ESXi is VXLAN encapsulated. When Netflow is enabled on the host uplink, the IP flow records are exported using a custom IPFIX flow-record template that includes the outer VXLAN UDP/IP header information as well as the information of the inner encapsulated IP packet. Such flow record thus provides visibility on the VTEP that is encapsulating the packet (outer header) and the details of the Virtual Machine that generated inter-host traffic (inner header) on a NSX Logical Switch (VXLAN).

Please refer to "Appendix B: IPFIX Templates" for more details on the IPFIX Templates for VDS

Enabling IPFIX

Access your vCenter using vSphere Web Client and browse to **Networking**

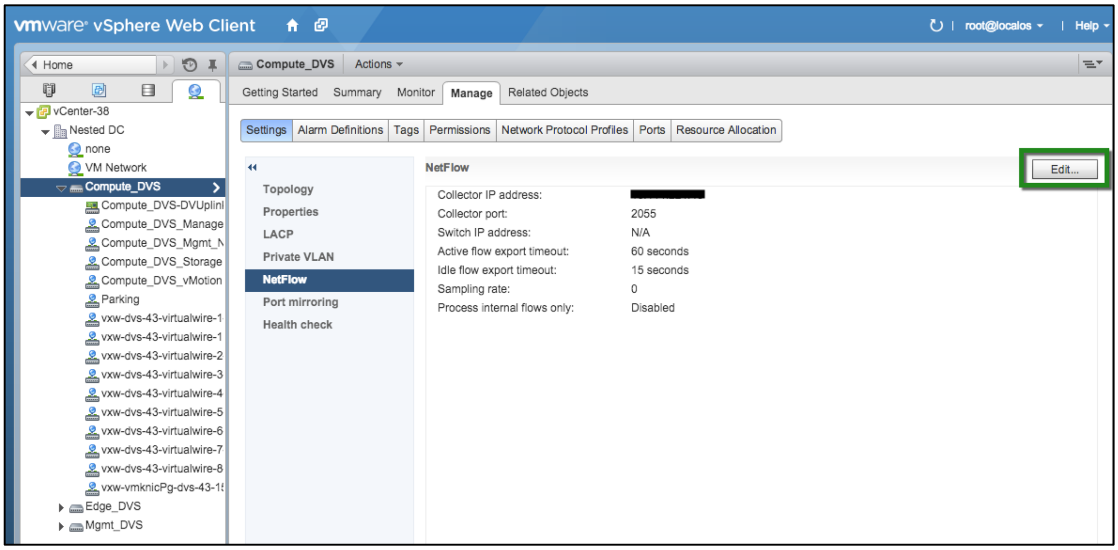


Select the VDS that is part of the Transport Zone. In the example below, we select "Compute_DVS". Browse to Manage → Settings → Netflow

Click on Edit to add Netflow Collector.

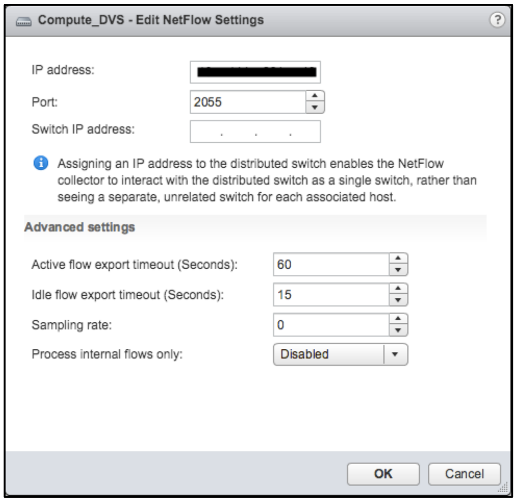
The following settings are configurable for a Netflow Collector:

- IP Address of Netflow collector (IPv4 address only)
- Port Number that the Netflow collector is listening on. Please refer to the documentation for your Netflow collector to match the port. IANA assigned port number for IPFIX is 4739.
- Switch IP Address: By configuring this field, all the ESXi hosts will source the IPFIX flow using this one single IP (for each VDS). On your Netflow Collector, only one exporter IP address will appear per VDS.
- Before we cover the Advanced Settings, let's describe the **Active flow export timeout**, **Idle flow export timeout** and **Sampling rate**. Typically, VDS will export the flow record maintained in its cache when the connection is terminated or the cache is full. However for long-lived connections (for example, a long duration ftp session), VDS will export the flow record every ***60* seconds** (default) even if the flow is still active. For flows that hit the idle flow export timeout (when the connection is inactive but not yet terminated) VDS will export the flow record every ***15 seconds***, by default. The sampling rate determines the data that is collected and exported by VDS. A sampling rate of "2" means VDS will collect data from one out of every 2 packets. Sampling rate of 0 (default) collects data from every packet.
- Select **Process internal flows only** to collect data only on network activity between virtual machines on the same host. In case we want to capture flows that are between VMs on different hosts, which will be the most likely scenario, we should leave this option to **Disabled**.



NetFlow

Collector IP address:	[redacted]
Collector port:	2055
Switch IP address:	N/A
Active flow export timeout:	60 seconds
Idle flow export timeout:	15 seconds
Sampling rate:	0
Process internal flows only:	Disabled



Compute_DVS - Edit NetFlow Settings

IP address:	[redacted]
Port:	2055
Switch IP address:	. . .

Assigning an IP address to the distributed switch enables the NetFlow collector to interact with the distributed switch as a single switch, rather than seeing a separate, unrelated switch for each associated host.

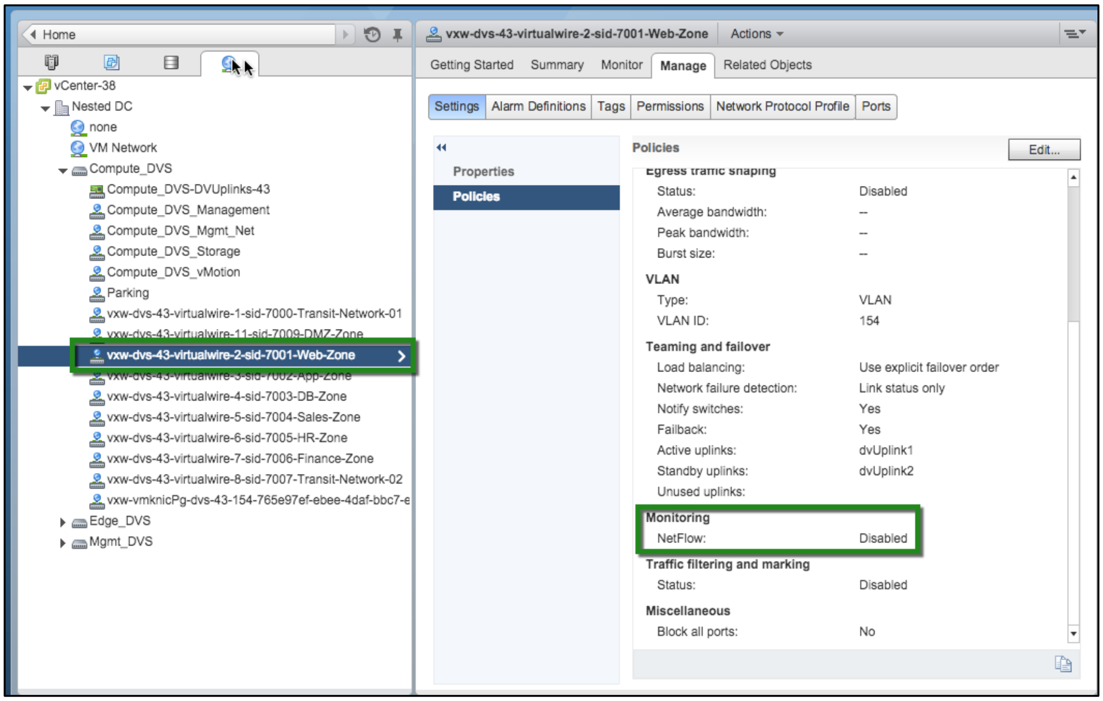
Advanced settings

Active flow export timeout (Seconds):	60
Idle flow export timeout (Seconds):	15
Sampling rate:	0
Process internal flows only:	Disabled

The next step is to enable Monitoring on a dvPortgroup to start sending flows to the Netflow collector.

Enabling Netflow on virtual dvPortgroup

To enable Netflow on a Logical Switch, browse to the VDS port-group that is backing the Logical Switch. In the example below, we have Logical Switch -- “**Web-Zone**” which is backed by VDS port-group – “**vxw-dvs-43-virtualwire-2-sid-7001-Web-Zone**” on Compute_DVS. The name of the VDS port-group in your environment may vary, however the naming convention is that the number **7001** represent the VXLAN ID that was assigned to this Logical Switch appended by the name of the Logical Switch.


Home
vCenter-38
Nested DC
none
VM Network
Compute_DVS
Compute_DVS-DVUplinks-43
Compute_DVS_Management
Compute_DVS_Mgmt_Net
Compute_DVS_Storage
Compute_DVS_vMotion
Parking
vxw-dvs-43-virtualwire-1-sid-7000-Transit-Network-01
vxw-dvs-43-virtualwire-11-sid-7009-DMZ-Zone
vxw-dvs-43-virtualwire-2-sid-7001-Web-Zone
vxw-dvs-43-virtualwire-3-sid-7002-App-Zone
vxw-dvs-43-virtualwire-4-sid-7003-DB-Zone
vxw-dvs-43-virtualwire-5-sid-7004-Sales-Zone
vxw-dvs-43-virtualwire-6-sid-7005-HR-Zone
vxw-dvs-43-virtualwire-7-sid-7006-Finance-Zone
vxw-dvs-43-virtualwire-8-sid-7007-Transit-Network-02
vxw-vmknicPg-dvs-43-154-765e97ef-ebee-4daf-bbc7-e
Edge_DVS
Mgmt_DVS
vxw-dvs-43-virtualwire-2-sid-7001-Web-Zone
Actions
Getting Started
Summary
Monitor
Manage
Related Objects
Settings
Alarm Definitions
Tags
Permissions
Network Protocol Profile
Ports
Properties
Policies
Edit...

Policies

Egress traffic shaping

Status: Disabled

Average bandwidth: --

Peak bandwidth: --

Burst size: --

VLAN

Type: VLAN

VLAN ID: 154

Teaming and failover

Load balancing: Use explicit failover order

Network failure detection: Link status only

Notify switches: Yes

Failback: Yes

Active uplinks: dvUplink1

Standby uplinks: dvUplink2

Unused uplinks:

Monitoring

NetFlow: Disabled

Traffic filtering and marking

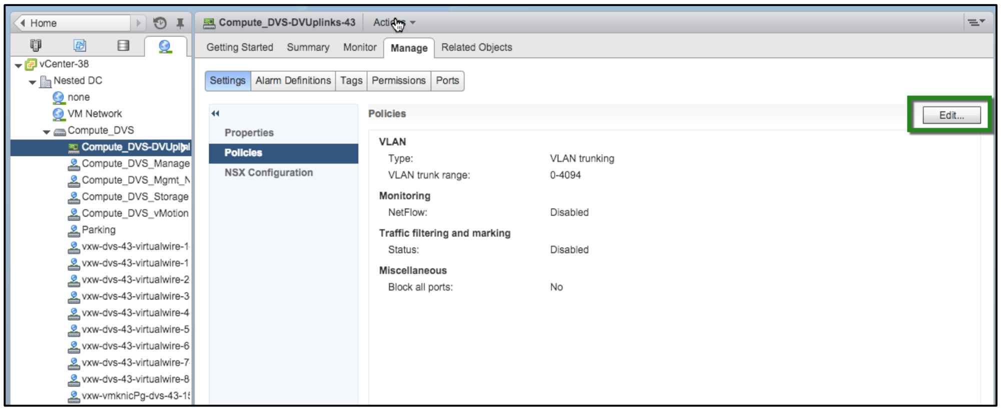
Status: Disabled

Miscellaneous

Block all ports: No

Enabling Netflow on dvUplink

The steps to enable Netflow on the dvUplink are similar to virtual dvPortGroup. The screenshots of the procedure is shown below.



Policies

VLAN

Type: VLAN trunking

VLAN trunk range: 0-4094

Monitoring

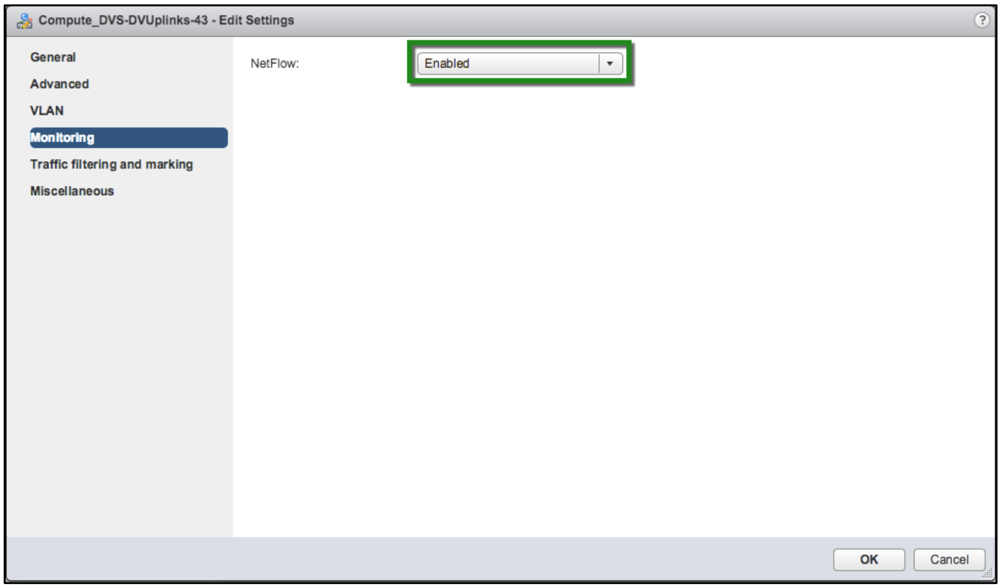
NetFlow: Disabled

Traffic filtering and marking

Status: Disabled

Miscellaneous

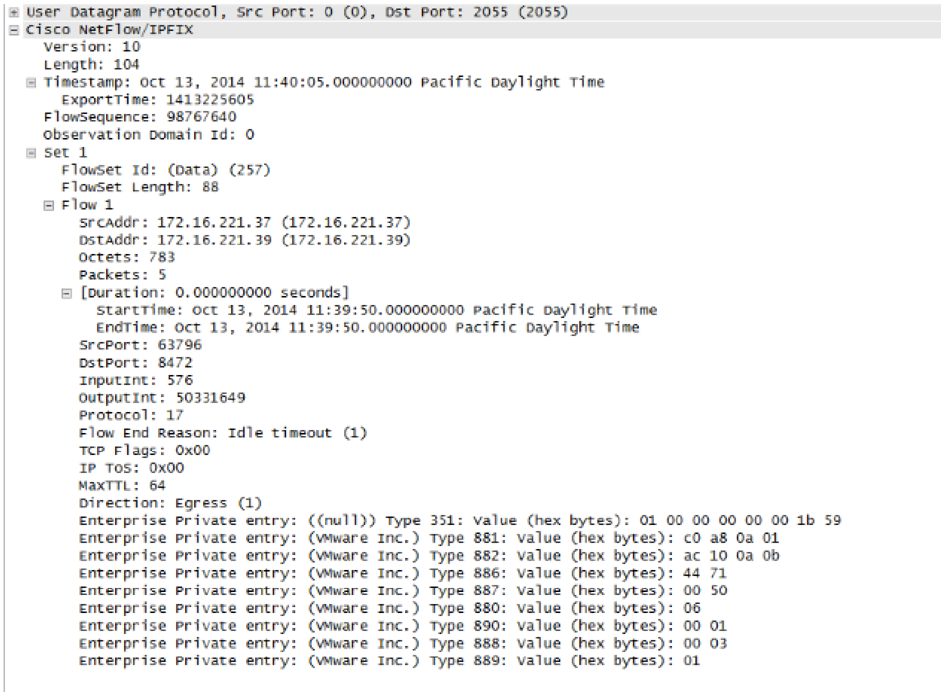
Block all ports: No



Compute_DVS-DVUplinks-43 - Edit Settings

NetFlow: Enabled

Sample VDS IPFIX record captured using Wireshark tool.



```
⊞ User Datagram Protocol, Src Port: 0 (0), Dst Port: 2055 (2055)
⊟ Cisco NetFlow/IPFIX
    Version: 10
    Length: 104
  ⊟ Timestamp: Oct 13, 2014 11:40:05.000000000 Pacific Daylight Time
      ExportTime: 1413225605
    FlowSequence: 98767640
    Observation Domain Id: 0
  ⊟ Set 1
      FlowSet Id: (Data) (257)
      FlowSet Length: 88
    ⊟ Flow 1
        SrcAddr: 172.16.221.37 (172.16.221.37)
        DstAddr: 172.16.221.39 (172.16.221.39)
        Octets: 783
        Packets: 5
      ⊟ [Duration: 0.000000000 seconds]
          StartTime: Oct 13, 2014 11:39:50.000000000 Pacific Daylight Time
          EndTime: Oct 13, 2014 11:39:50.000000000 Pacific Daylight Time
        SrcPort: 63796
        DstPort: 8472
        InputInt: 576
        OutputInt: 50331649
        Protocol: 17
        Flow End Reason: Idle timeout (1)
        TCP Flags: 0x00
        IP ToS: 0x00
        MaxTTL: 64
        Direction: Egress (1)
        Enterprise Private entry: ((null)) Type 351: Value (hex bytes): 01 00 00 00 00 00 1b 59
        Enterprise Private entry: (VMware Inc.) Type 881: Value (hex bytes): c0 a8 0a 01
        Enterprise Private entry: (VMware Inc.) Type 882: Value (hex bytes): ac 10 0a 0b
        Enterprise Private entry: (VMware Inc.) Type 886: Value (hex bytes): 44 71
        Enterprise Private entry: (VMware Inc.) Type 887: Value (hex bytes): 00 50
        Enterprise Private entry: (VMware Inc.) Type 880: Value (hex bytes): 06
        Enterprise Private entry: (VMware Inc.) Type 890: Value (hex bytes): 00 01
        Enterprise Private entry: (VMware Inc.) Type 888: Value (hex bytes): 00 03
        Enterprise Private entry: (VMware Inc.) Type 889: Value (hex bytes): 01
```

IPFIX on DFW

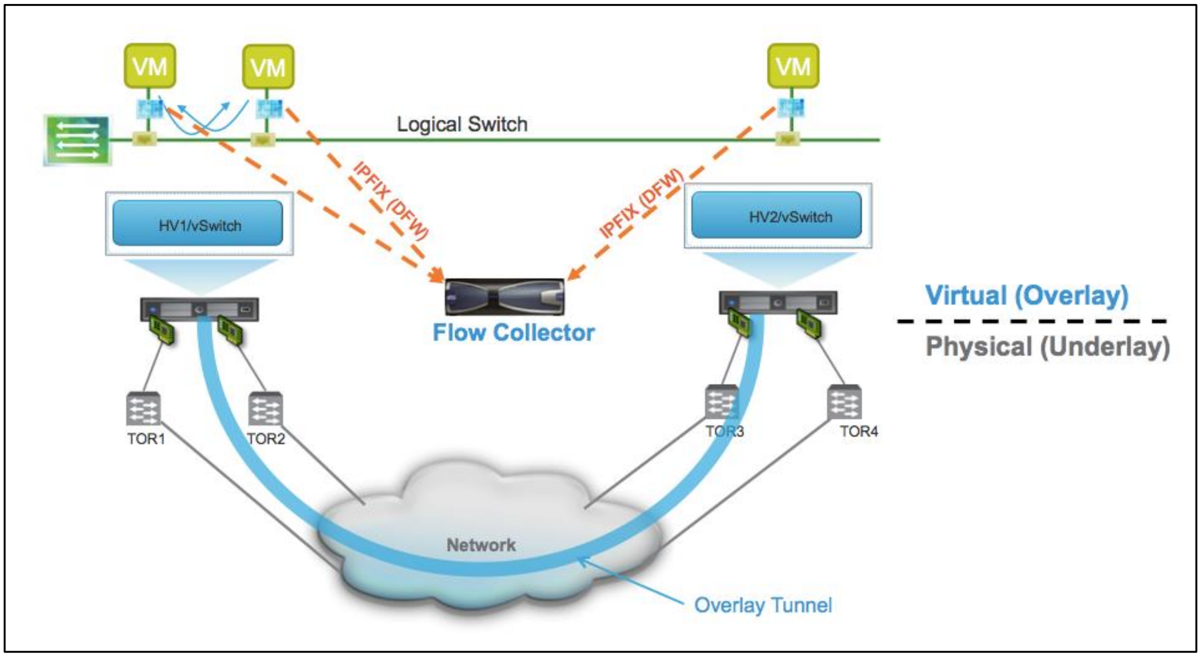


Figure: IPFIX from DFW

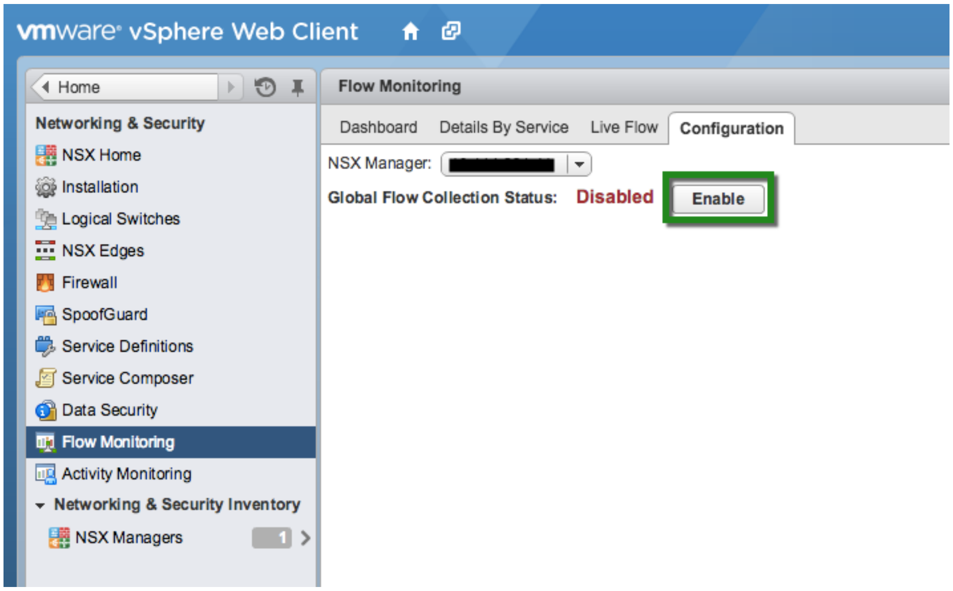
In the previous section we covered IPFIX support on VMware Distributed Switch. This section describes IPFIX support for the Distributed Firewall (DFW).

The distributed firewall implements stateful tracking of flows and the tracked flows go through a set of state changes. IPFIX can be used to export data about the status of a flow. The tracked events include flow creation, flow denial, flow update and flow teardown.

Please refer to "Appendix B: IPFIX Templates" for more details on the IPFIX Templates for VDS.

Steps to enable IPFIX flow export for DFW involves enabling Global Flow Collection, Enabling IPFIX flow export and setting up IPFIX Collectors.

i) Browse to NSX Home → Flow Monitoring. Click **Enable** for Global Flow Collection Status.

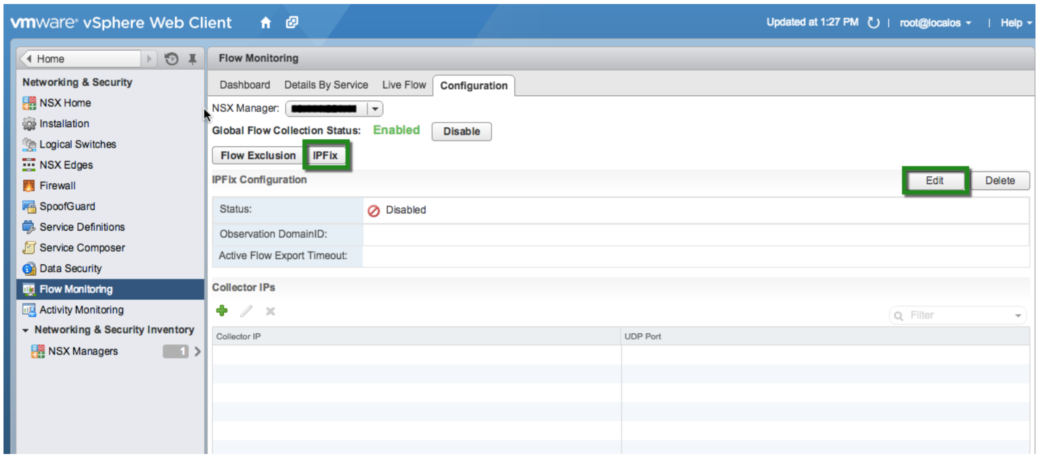


Flow Monitoring

NSX Manager: [redacted]

Global Flow Collection Status: Disabled

ii) Once you click Enable Flow Collection, a new tab will appear for IPFIX. Browse to the IPFIX tab and then click Edit to enable IPFIX flow export.



Flow Monitoring

NSX Manager: [redacted]

Global Flow Collection Status: Enabled

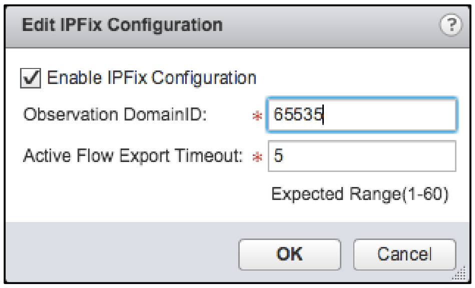
IPFix Configuration

Status:	Disabled
Observation DomainID:	
Active Flow Export Timeout:	

Collector IPs

Collector IP	UDP Port

iii) Enter **Observation DomainID** (valid range 0-65535) and **Active Flow Export Timeout** (in minutes). Click OK.



Edit IPFix Configuration

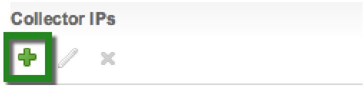
☑ Enable IPFix Configuration

Observation DomainID: * 65535

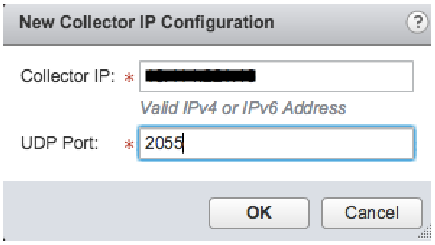
Active Flow Export Timeout: * 5

Expected Range(1-60)

iv) To add a new Collector, click on the + symbol.



Provide Collector IP and UDP Port number (please refer to your Netflow collector documentation to determine the port number).



New Collector IP Configuration

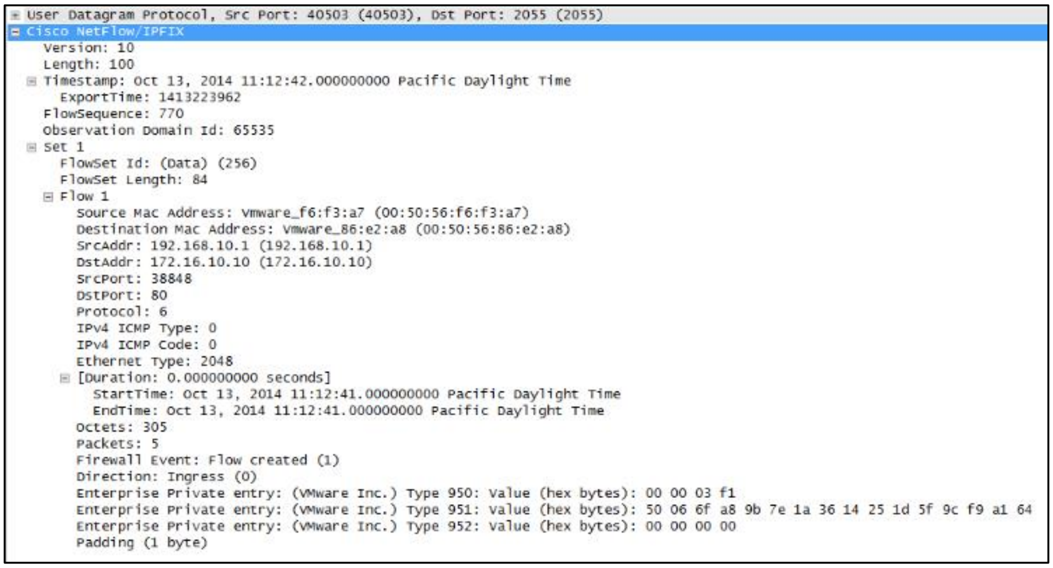
Collector IP: * [redacted]

Valid IPv4 or IPv6 Address

UDP Port: * 2055

v) Click **Publish Changes** to bring it into effect immediately.

Sample DFW IPFIX flow record captured using Wireshark tool.



```
⊞ User Datagram Protocol, Src Port: 40503 (40503), Dst Port: 2055 (2055)
⊟ Cisco NetFlow/IPFIX
    Version: 10
    Length: 100
  ⊟ Timestamp: Oct 13, 2014 11:12:42.000000000 Pacific Daylight Time
      ExportTime: 1413223962
    FlowSequence: 770
    Observation Domain Id: 65535
  ⊟ Set 1
      FlowSet Id: (Data) (256)
      FlowSet Length: 84
    ⊟ Flow 1
        Source Mac Address: Vmware_f6:f3:a7 (00:50:56:f6:f3:a7)
        Destination Mac Address: Vmware_86:e2:a8 (00:50:56:86:e2:a8)
        SrcAddr: 192.168.10.1 (192.168.10.1)
        DstAddr: 172.16.10.10 (172.16.10.10)
        SrcPort: 38848
        DstPort: 80
        Protocol: 6
        IPv4 ICMP Type: 0
        IPv4 ICMP Code: 0
        Ethernet Type: 2048
      ⊟ [Duration: 0.000000000 seconds]
          StartTime: Oct 13, 2014 11:12:41.000000000 Pacific Daylight Time
          EndTime: Oct 13, 2014 11:12:41.000000000 Pacific Daylight Time
        Octets: 305
        Packets: 5
        Firewall Event: Flow created (1)
        Direction: Ingress (0)
        Enterprise Private entry: (VMware Inc.) Type 950: Value (hex bytes): 00 00 03 f1
        Enterprise Private entry: (VMware Inc.) Type 951: Value (hex bytes): 50 06 6f a8 9b 7e 1a 36 14 25 1d 5f 9c f9 a1 64
        Enterprise Private entry: (VMware Inc.) Type 952: Value (hex bytes): 00 00 00 00
        Padding (1 byte)
```

Packet Visibility

Network troubleshooting and monitoring tools are critical in any environment, particularly in virtual datacenter environment where you have many applications or workloads consolidated on server virtualization platforms such as vSphere. VMware vSphere Distributed Switch (VDS) supports industry standard features such as port mirroring and Netflow that provides traffic visibility to the administrator. We covered Netflow (IPFIX) in the previous section. In this section we will cover SPAN and Remote SPAN feature.

SPAN on Virtual Switch

VMware vSphere Distributed Switch (VDS) supports mirroring traffic (SPAN) between two virtual ports on a VDS. This feature is only supported when the source and the destination virtual ports are on the same host. This feature is useful when you have a Service VM such as Intrusion Detection System (IDS), that monitors the mirrored traffic for a set of "protected" Virtual Machines, running on the same host. If you wish to mirror traffic from a hypervisor host to either a virtual machine running on a different host or to a physical host reachable over IP network, RSPAN or L3SPAN will be more suited. Please refer to next section for more details on RSPAN and L3SPAN capability.

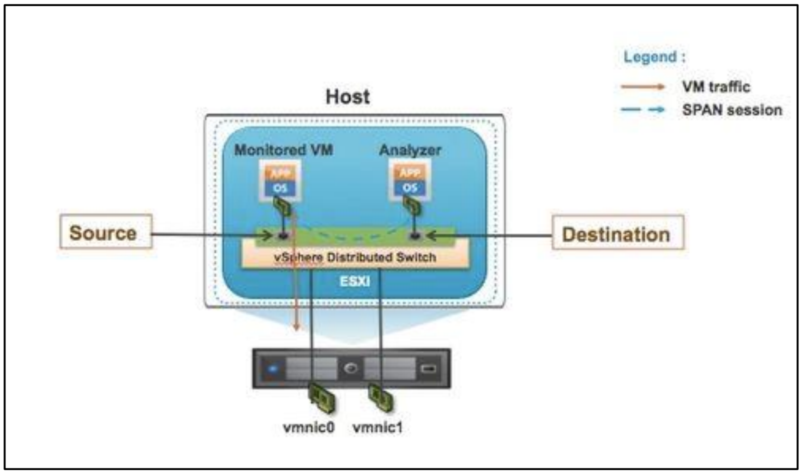
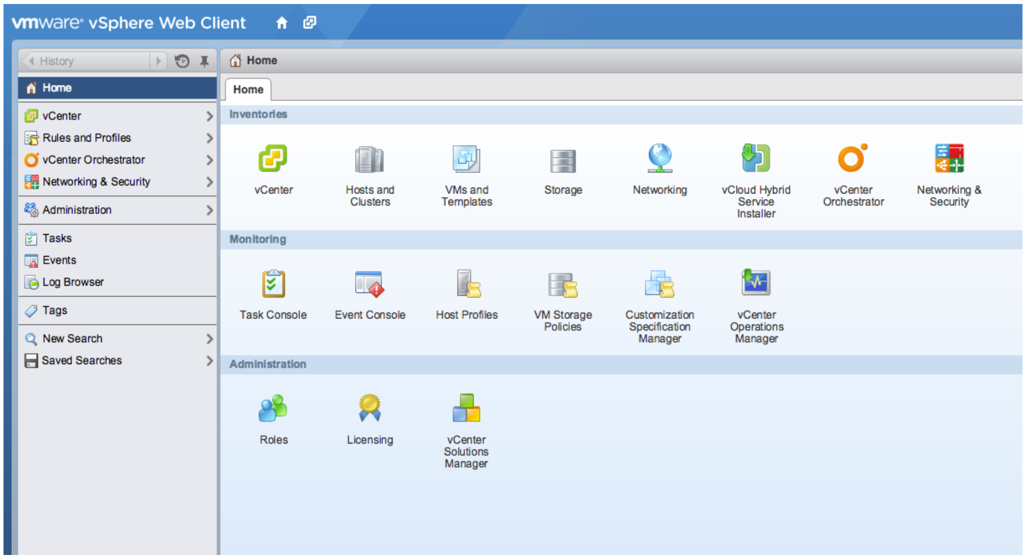


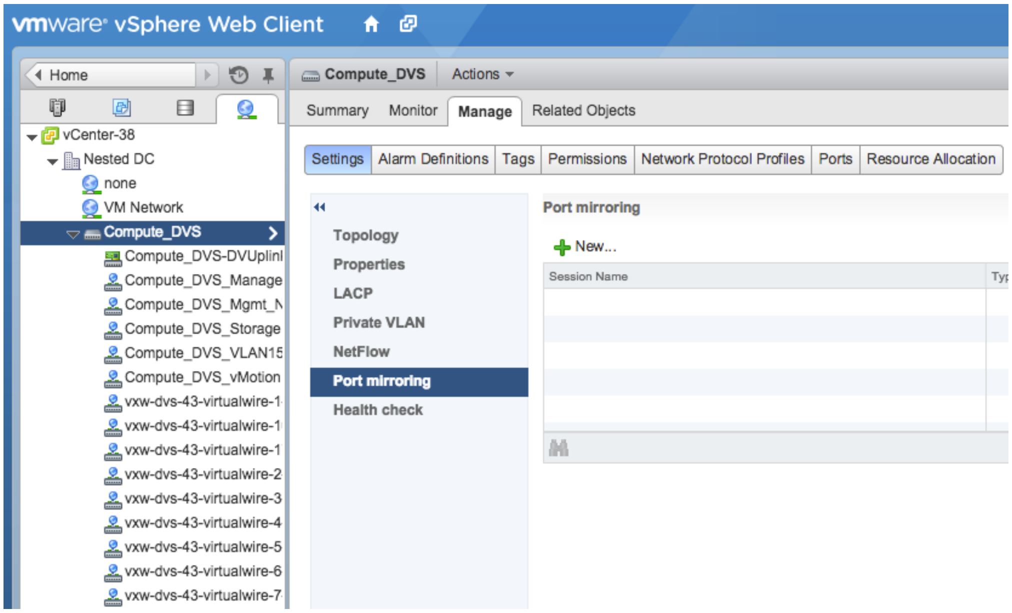
Figure: Monitoring VM traffic using SPAN.

To enable SPAN on the VDS:

Step 1) On your vSphere Web Client, click on Home -> Networking.



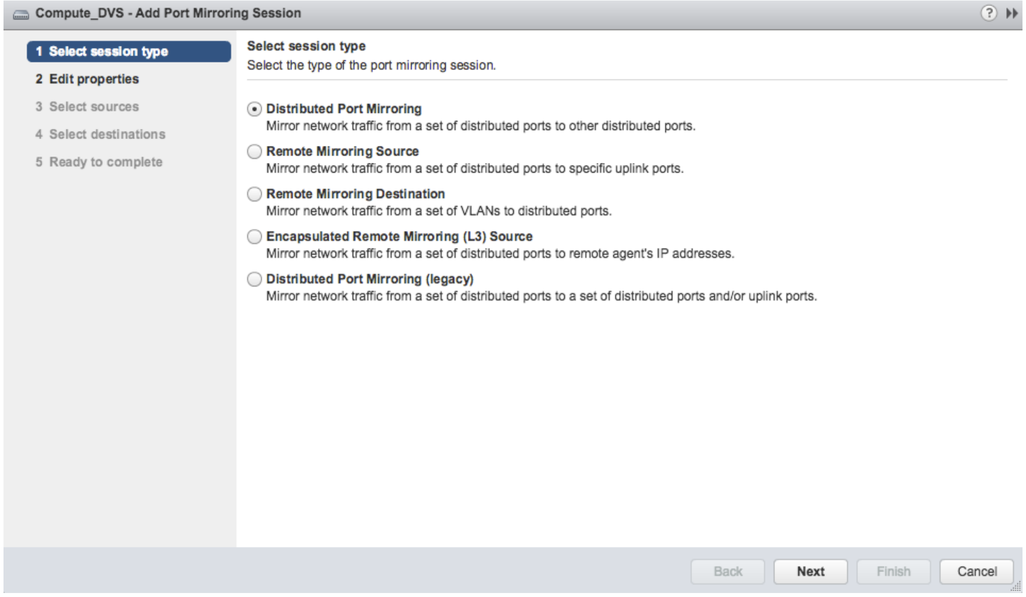
Step 2) Select the VDS that is hosting the VM. Click on the Manage tab.



Step 3) Click Settings → Port mirroring to configuring SPAN session. If you have previously configured any Port mirroring sessions, those will be listed here. You can edit any of them from the list by selecting it and clicking Edit.

Step 4) To create a new SPAN session, click New. This will bring up a wizard to configure a new session.

Step 5) For SPAN, Select the session type as “Distributed Port Mirroring” as shown in the below figure. Click Next.



Compute_DVS - Add Port Mirroring Session

Select session type

Select the type of the port mirroring session.

● Distributed Port Mirroring
Mirror network traffic from a set of distributed ports to other distributed ports.

○ Remote Mirroring Source
Mirror network traffic from a set of distributed ports to specific uplink ports.

○ Remote Mirroring Destination
Mirror network traffic from a set of VLANs to distributed ports.

○ Encapsulated Remote Mirroring (L3) Source
Mirror network traffic from a set of distributed ports to remote agent's IP addresses.

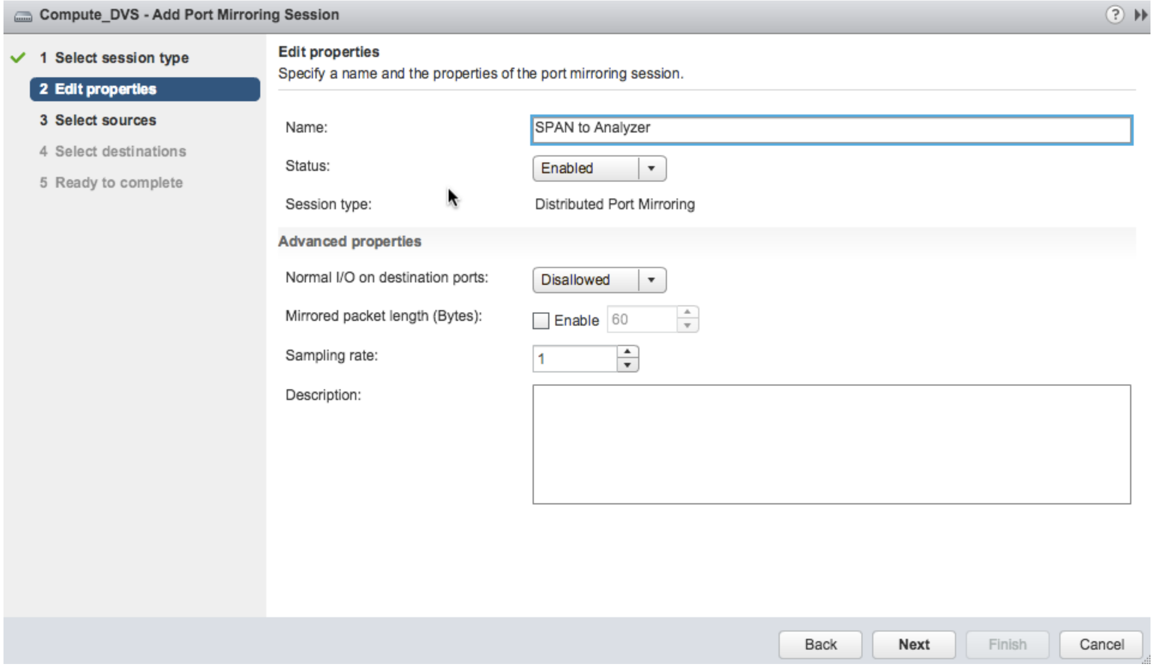
○ Distributed Port Mirroring (legacy)
Mirror network traffic from a set of distributed ports to a set of distributed ports and/or uplink ports.

Step 6) Key configuration parameters are:

- Status: Enabled/Disabled. Enables user to disable or enable a mirroring session without having to delete the mirroring configuration.
- Normal I/O on destination ports: Allowing normal I/O on destination port allows the destination virtual port to communicate normally.

  Set this to “Disallowed” when you have a dedicated traffic-collecting VM interface. Example: In case of IDS application (or similar), typically there will be a dedicated out-of-band Management interface and a separate Traffic-sensing interface. In such a case, you may select “Disallowed” option for the Traffic-sensing VM interface. In case the Management traffic and mirrored traffic share a common virtual interface, then select “Allowed” option.
- Mirrored packet length (Bytes): This option will only copy first X bytes of the packet to the destination port.
- Sampling rate: 1 means every packet will be mirrored, 2 means 1 in every 2 packets will be mirrored and so on.

Click Next to Proceed to selecting the VMs or Ports from where you would like to mirror traffic.



Compute_DVS - Add Port Mirroring Session

Edit properties

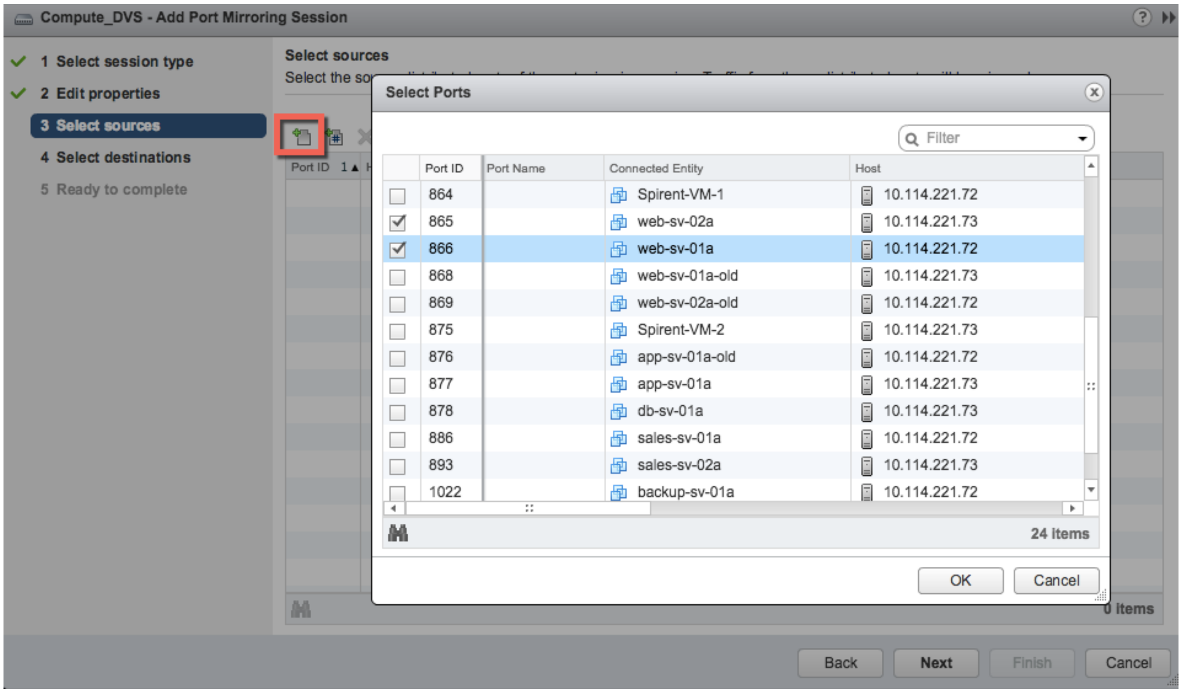
Specify a name and the properties of the port mirroring session.

Name:	SPAN to Analyzer
Status:	Enabled
Session type:	Distributed Port Mirroring

Advanced properties

Normal I/O on destination ports:	Disallowed
Mirrored packet length (Bytes):	□ Enable 60
Sampling rate:	1
Description:	

Step 7) In this step, you can select the source vnic(s) that will be mirrored. This can be performed by clicking the button.



Compute_DVS - Add Port Mirroring Session

Select sources

Select the so

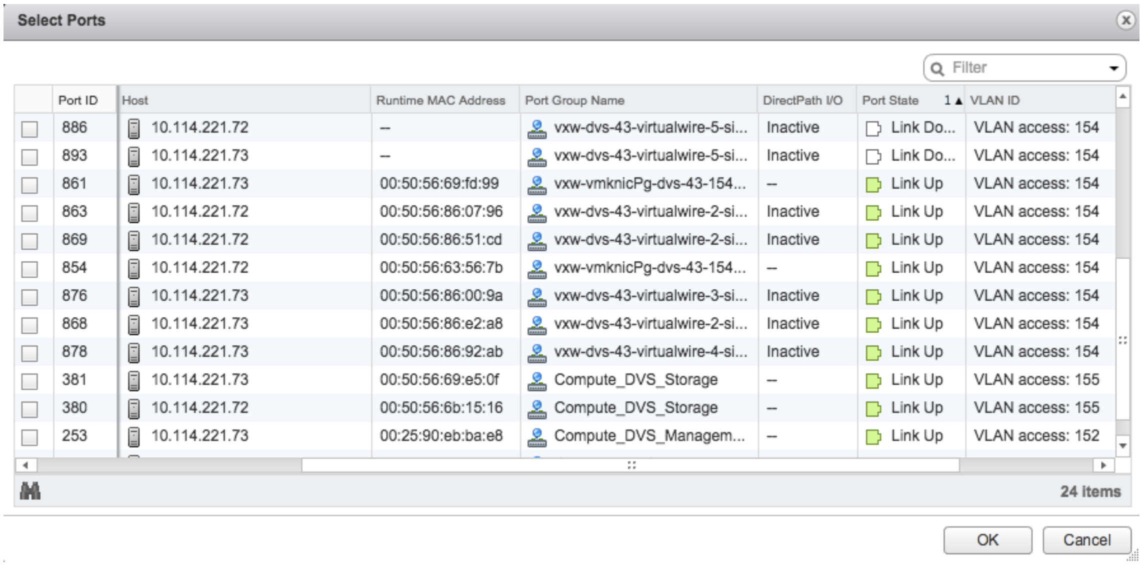
Select Ports

	Port ID	Port Name	Connected Entity	Host
□	864		Spirent-VM-1	10.114.221.72
☑	865		web-sv-02a	10.114.221.73
☑	866		web-sv-01a	10.114.221.72
□	868		web-sv-01a-old	10.114.221.73
□	869		web-sv-02a-old	10.114.221.72
□	875		Spirent-VM-2	10.114.221.73
□	876		app-sv-01a-old	10.114.221.72
□	877		app-sv-01a	10.114.221.73
□	878		db-sv-01a	10.114.221.73
□	886		sales-sv-01a	10.114.221.72
□	893		sales-sv-02a	10.114.221.73
□	1022		backup-sv-01a	10.114.221.72

24 items

This will pop-up a Select Ports menu where you can select the VM's vnics that must be mirrored. You can select multiple vNics belonging to same or different VMs.

[Note: If the VMs have multiple NICs, then you may want to scroll to the right to check the MAC address or Port Group Name, in order to identify the correct vNIC that must be mirrored. See figure below]



Select Ports

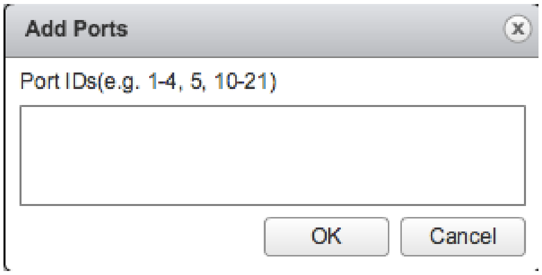
Filter

	Port ID	Host	Runtime MAC Address	Port Group Name	DirectPath I/O	Port State 1 ▲	VLAN ID
□	886	10.114.221.72	--	vxw-dvs-43-virtualwire-5-si...	Inactive	Link Do...	VLAN access: 154
□	893	10.114.221.73	--	vxw-dvs-43-virtualwire-5-si...	Inactive	Link Do...	VLAN access: 154
□	861	10.114.221.73	00:50:56:69:fd:99	vxw-vmknicPg-dvs-43-154...	--	Link Up	VLAN access: 154
□	863	10.114.221.72	00:50:56:86:07:96	vxw-dvs-43-virtualwire-2-si...	Inactive	Link Up	VLAN access: 154
□	869	10.114.221.72	00:50:56:86:51:cd	vxw-dvs-43-virtualwire-2-si...	Inactive	Link Up	VLAN access: 154
□	854	10.114.221.72	00:50:56:63:56:7b	vxw-vmknicPg-dvs-43-154...	--	Link Up	VLAN access: 154
□	876	10.114.221.73	00:50:56:86:00:9a	vxw-dvs-43-virtualwire-3-si...	Inactive	Link Up	VLAN access: 154
□	868	10.114.221.73	00:50:56:86:e2:a8	vxw-dvs-43-virtualwire-2-si...	Inactive	Link Up	VLAN access: 154
□	878	10.114.221.73	00:50:56:86:92:ab	vxw-dvs-43-virtualwire-4-si...	Inactive	Link Up	VLAN access: 154
□	381	10.114.221.73	00:50:56:69:e5:0f	Compute_DVS_Storage	--	Link Up	VLAN access: 155
□	380	10.114.221.72	00:50:56:6b:15:16	Compute_DVS_Storage	--	Link Up	VLAN access: 155
□	253	10.114.221.73	00:25:90:eb:ba:e8	Compute_DVS_Managem...	--	Link Up	VLAN access: 152

24 items

Click OK to save the selection.

[Note: Alternately, if you know the Port ID corresponding to the vNICs that must be mirrored, click the button to specify those source ports.]



Add Ports

Port IDs(e.g. 1-4, 5, 10-21)

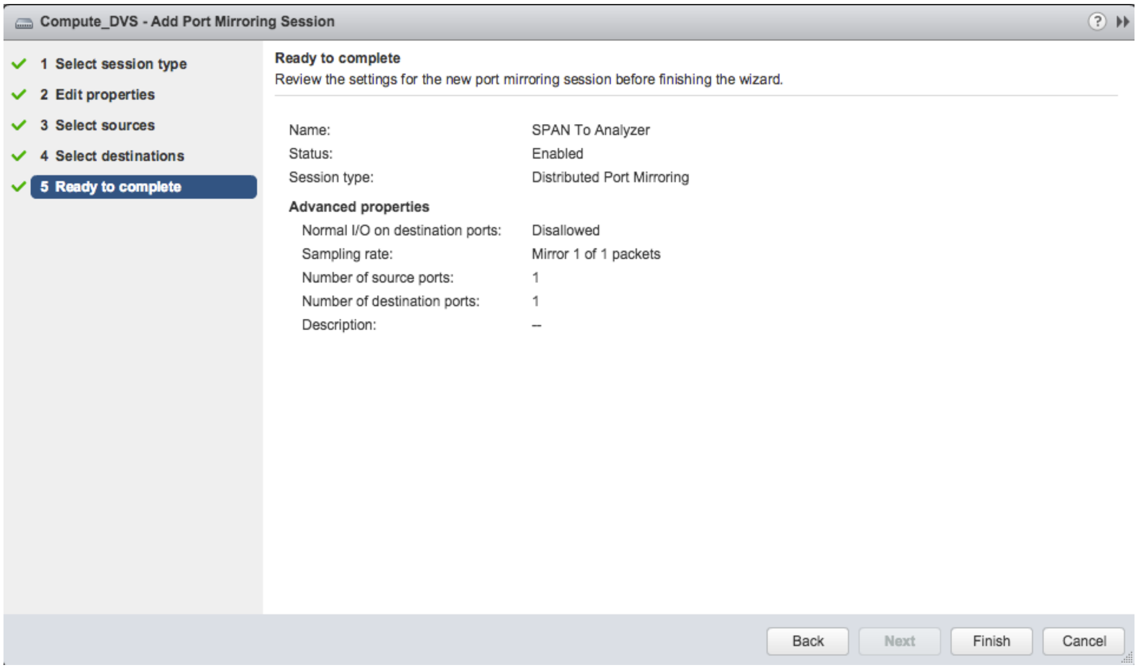
Step 8) Once you have selected the vNics, select the direction of the traffic that must be mirrored. By default, both Ingress and Egress traffic will be mirrored on that vNIC. If you wish to only mirror Ingress or Egress, then use these buttons to toggle the configuration.

Note: Ingress = Traffic exiting the VM's vNIC and **Ingressing** the vSwitch port.

Egress = Traffic entering the VM's vNIC and **Egressing** the vSwitch port.

Step 9) In this step, select the destination VM(s) or VDS port(s) that should receive the mirrored traffic. Similar to source port selections, you select the destination ports using the button (using VM's vnic) or the button (using VDS Port ID). Click Next to proceed.

Step 10) In this step, the summary of the SPAN session will be displayed. Click Finish to submit the changes.



Compute_DVS - Add Port Mirroring Session

Ready to complete

Review the settings for the new port mirroring session before finishing the wizard.

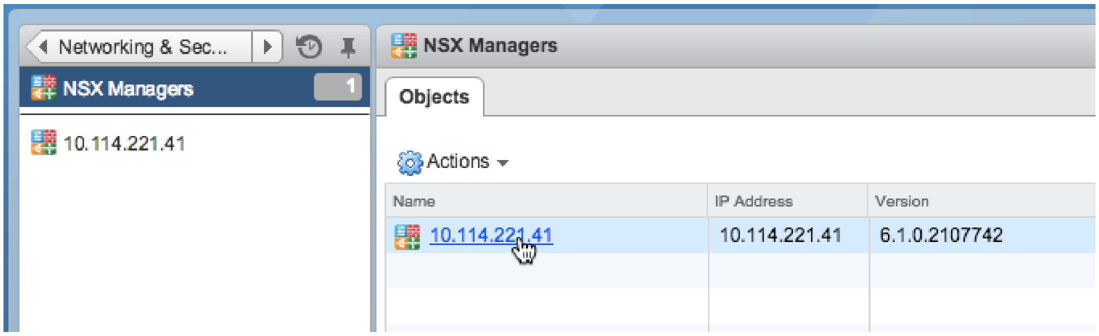
Name: SPAN To Analyzer
Status: Enabled
Session type: Distributed Port Mirroring

Advanced properties

Normal I/O on destination ports: Disallowed
Sampling rate: Mirror 1 of 1 packets
Number of source ports: 1
Number of destination ports: 1
Description: –

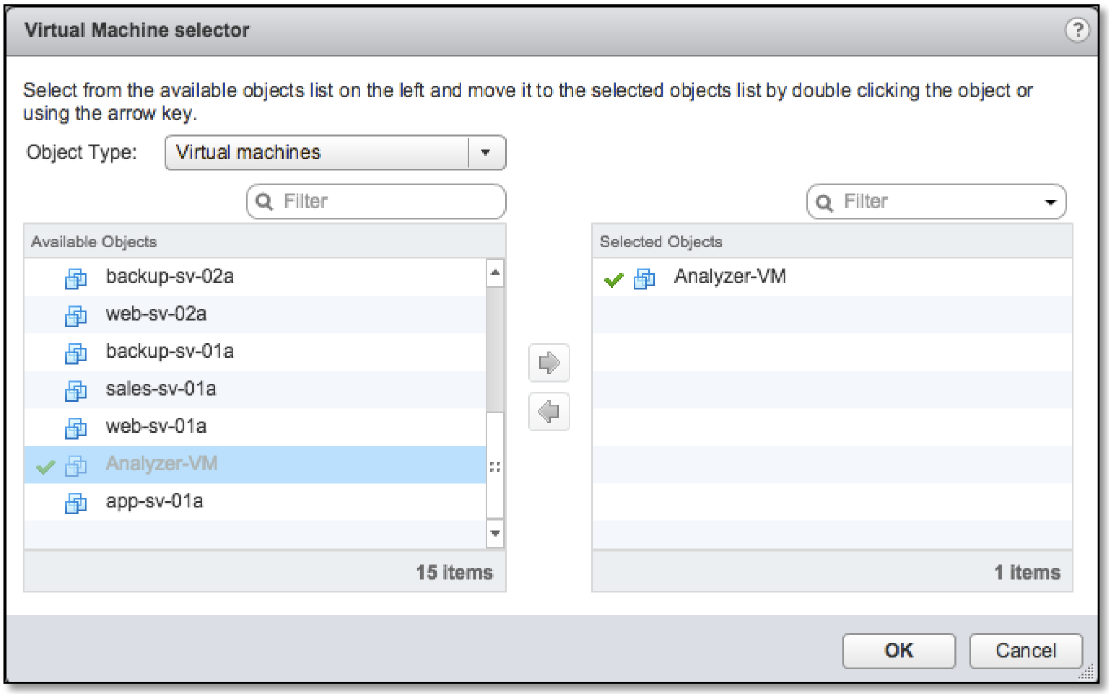
Step 11) In this final step, we will add the Analyzer VM in the exclusion list for DFW rules/dvfilters.

Click NSX Home → NSX Managers and then Select the NSX Manager by clicking on the IP address of your NSX Manager.



Name	IP Address	Version
10.114.221.41	10.114.221.41	6.1.0.2107742

Then Click on the **Manage** Tab → **Exclusion List**. Click ✚ to add the Analyzer VM to the exclusion list. Click OK to save the settings.



Virtual Machine selector

Select from the available objects list on the left and move it to the selected objects list by double clicking the object or using the arrow key.

Object Type: Virtual machines

Available Objects

- backup-sv-02a
- web-sv-02a
- backup-sv-01a
- sales-sv-01a
- web-sv-01a
- ✓ Analyzer-VM
- app-sv-01a

15 items

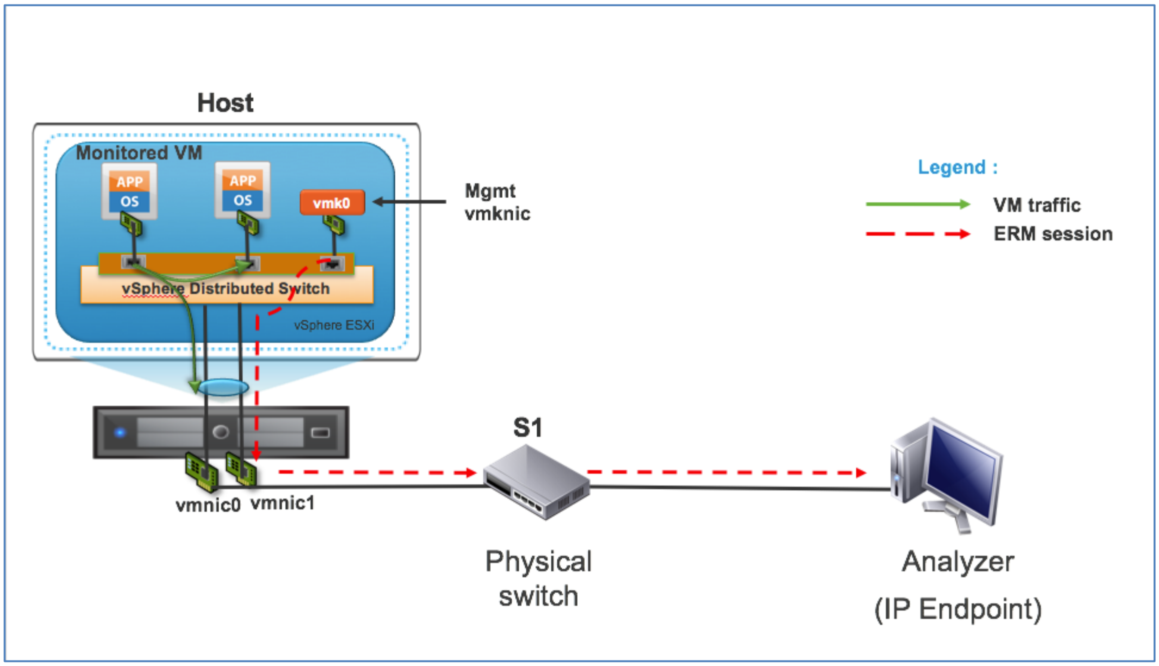
Selected Objects

- ✓ Analyzer-VM

1 items

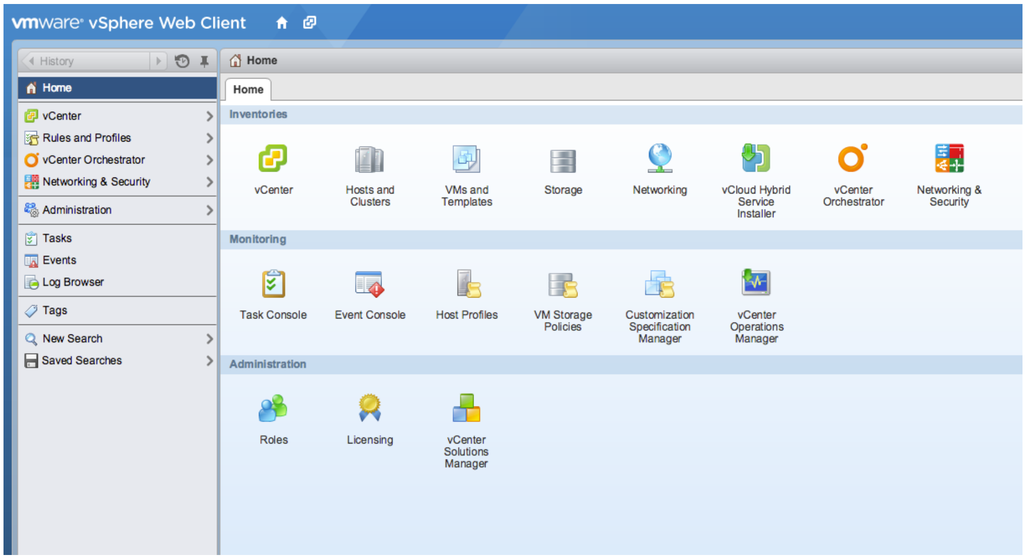
Encapsulated Remote Mirroring (ERM)

In the previous section, we discussed mirroring VM traffic to an Analyzer VM running on the same host and connected to the same VMware Distributed Switch. The VDS also supports the ability to mirror traffic for a Virtual Machine and send it to a remote Analyzer using Encapsulated Remote Mirroring (ERM). As shown in the figure, the Analyzer is a physical host that can be reached over IP. The ESXi host running the monitored VM mirrors the packets and encapsulates them as an IP packet (using GRE encapsulation) and then forwards it to the specified destination address. The mirrored traffic is always sourced from the Management vmkernel NIC of the host. The uplink teaming policy for VDS Port-Group for Management traffic must be reviewed to ensure that the mirror traffic will not overwhelm the uplinks. This is especially important if Management traffic is segregated onto separate slower 1Gig uplinks.

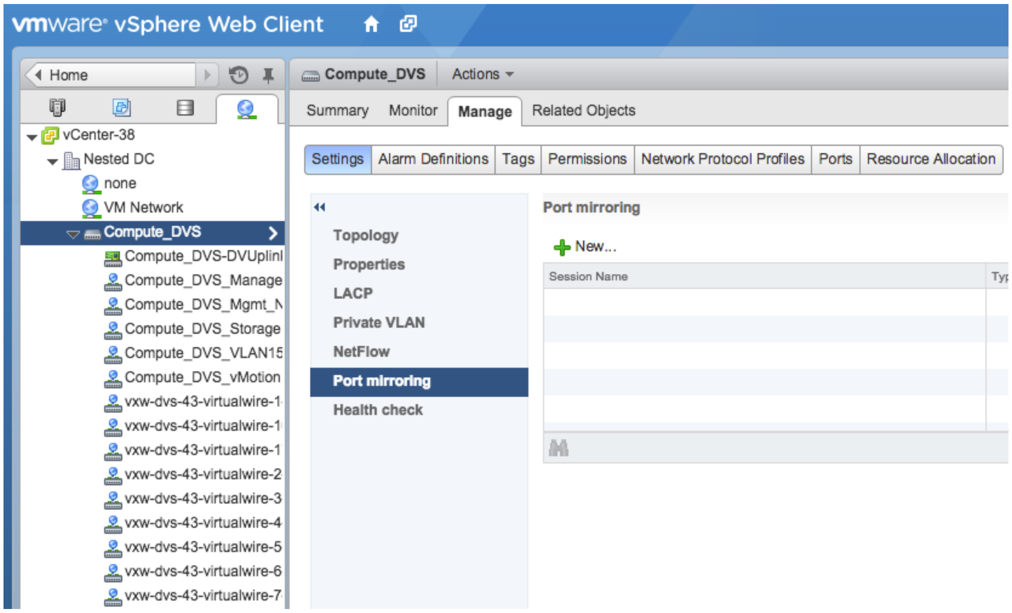


To enable Encapsulated Remote Mirroring on VDS:

Step 1) On your vSphere Web Client, click on Home -> Networking.



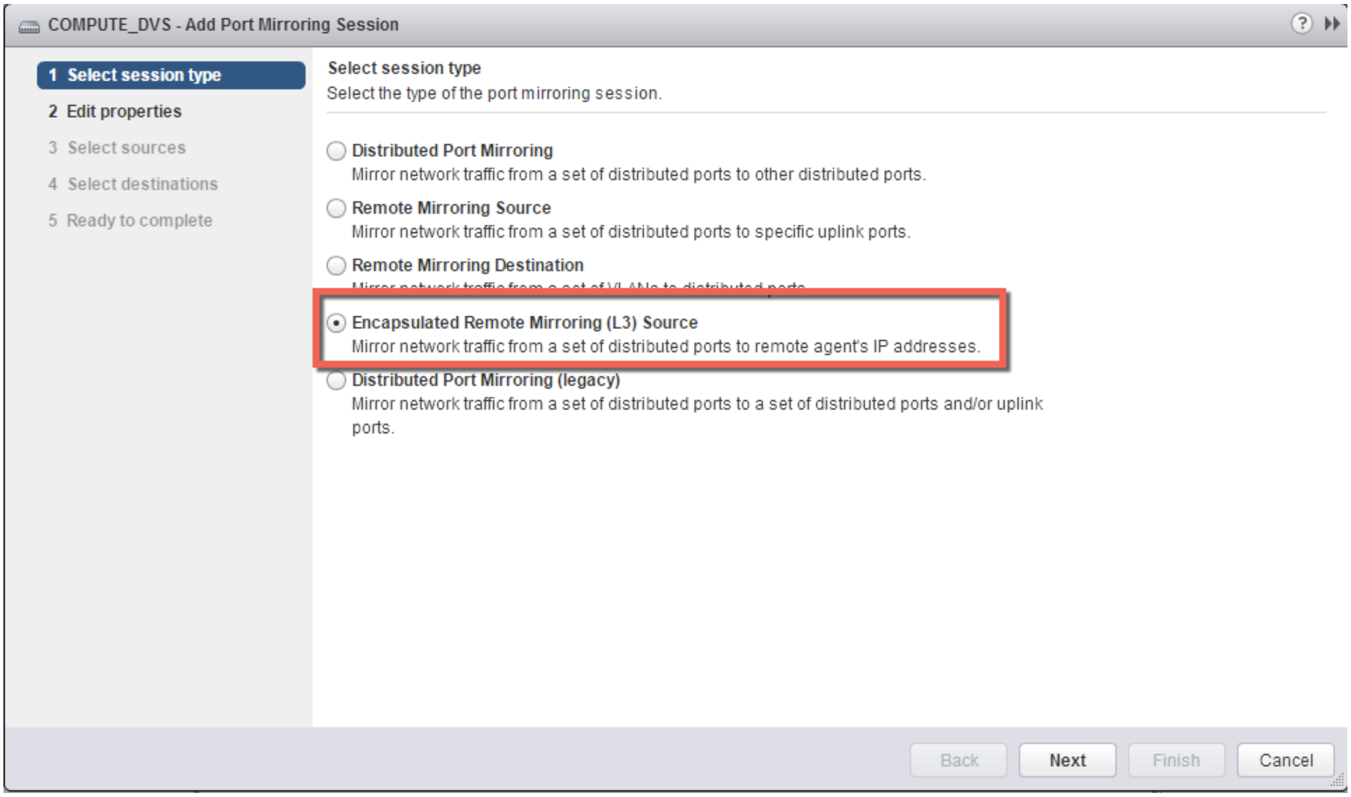
Step 2) Select the VDS that is hosting the VM. Click on the Manage tab.



Step 3) Click Settings → Port mirroring to configuring ERM session. If you have previously configured any Port mirroring sessions, those will be listed here. You can edit any of them from the list by selecting it and clicking Edit.

Step 4) To create a new ERM session, click New. This will bring up a wizard to configure a new session.

Step 5) For ERM, Select the session type as "Encapsulated Remote Mirroring" as shown in the below figure. Click Next.



Select session type

Select the type of the port mirroring session.

○ Distributed Port Mirroring
Mirror network traffic from a set of distributed ports to other distributed ports.

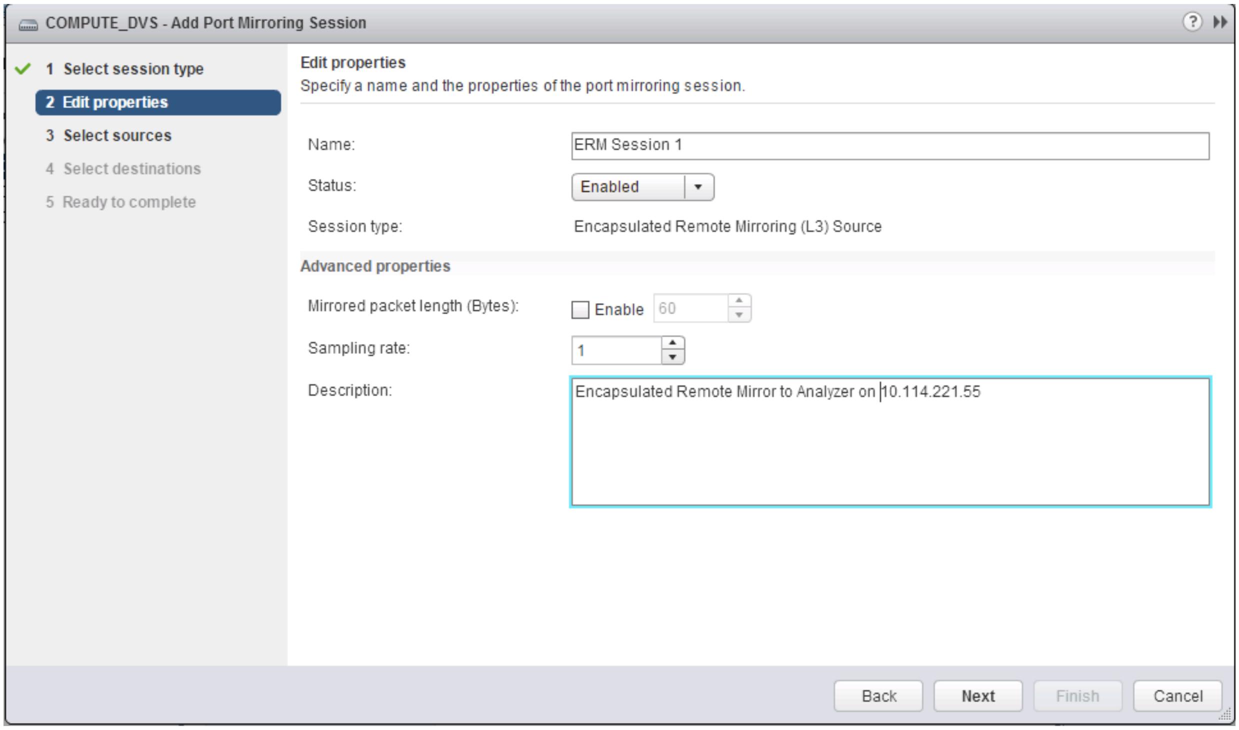
○ Remote Mirroring Source
Mirror network traffic from a set of distributed ports to specific uplink ports.

○ Remote Mirroring Destination
Mirror network traffic from a set of VLANs to distributed ports.

● Encapsulated Remote Mirroring (L3) Source
Mirror network traffic from a set of distributed ports to remote agent's IP addresses.

○ Distributed Port Mirroring (legacy)
Mirror network traffic from a set of distributed ports to a set of distributed ports and/or uplink ports.

Step 6) Provide a Name to the Session. Select Status as "Enabled" if you would like to start the capture immediately after you complete the configuration. Advanced properties such as Mirrored packet length and Sampling rate are optional and can be configured as per the requirement. Refer to previous section on SPAN for more details about these options. Click Next to select the Monitored VM(s).



Edit properties

Specify a name and the properties of the port mirroring session.

Name: ERM Session 1

Status: Enabled

Session type: Encapsulated Remote Mirroring (L3) Source

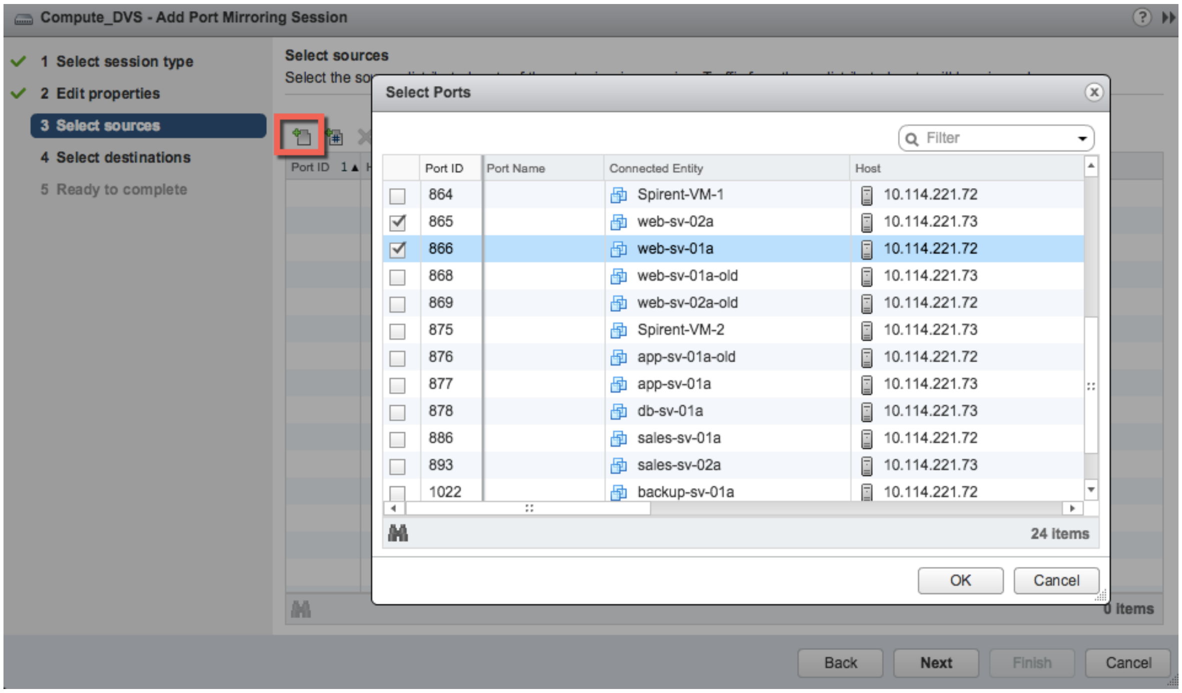
Advanced properties

Mirrored packet length (Bytes): □ Enable 60

Sampling rate: 1

Description: Encapsulated Remote Mirror to Analyzer on 10.114.221.55

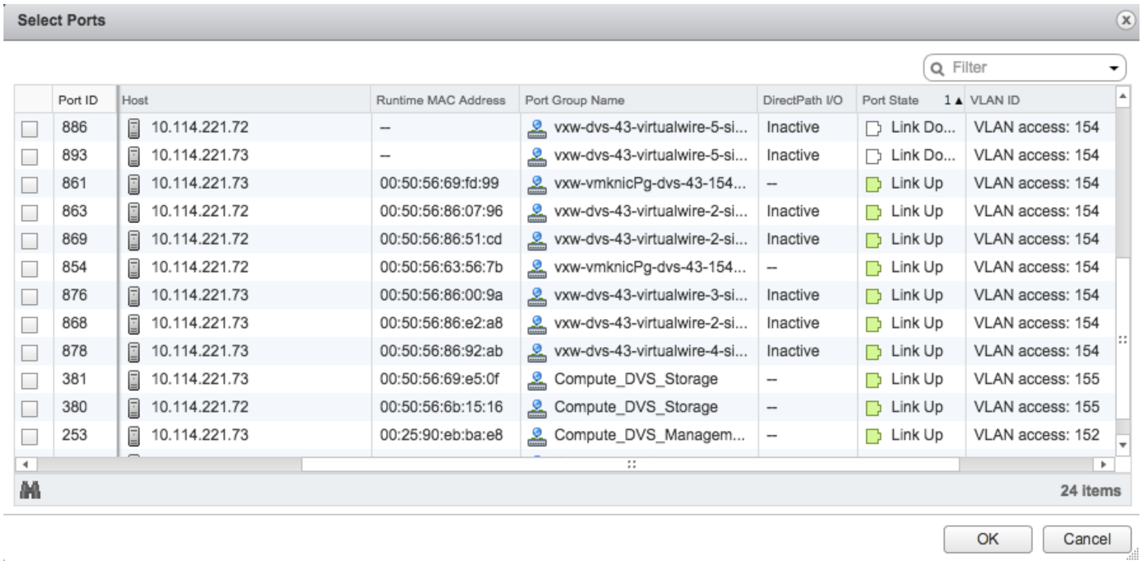
Step 7) In this step, you can select the source vnic(s) that will be mirrored. This can be performed by clicking the button.



	Port ID	Port Name	Connected Entity	Host
□	864		Spirent-VM-1	10.114.221.72
☑	865		web-sv-02a	10.114.221.73
☑	866		web-sv-01a	10.114.221.72
□	868		web-sv-01a-old	10.114.221.73
□	869		web-sv-02a-old	10.114.221.72
□	875		Spirent-VM-2	10.114.221.73
□	876		app-sv-01a-old	10.114.221.72
□	877		app-sv-01a	10.114.221.73
□	878		db-sv-01a	10.114.221.73
□	886		sales-sv-01a	10.114.221.72
□	893		sales-sv-02a	10.114.221.73
□	1022		backup-sv-01a	10.114.221.72

This will pop-up a Select Ports menu where you can select the VM's vnics that must be mirrored. You can select multiple vNics belonging to same or different VMs.

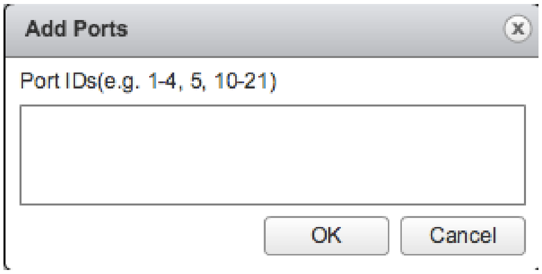
[Note: If the VMs have multiple NICs, then you may want to scroll to the right to check the MAC address or Port Group Name, in order to identify the correct vNIC that must be mirrored. See figure below]



	Port ID	Host	Runtime MAC Address	Port Group Name	DirectPath I/O	Port State 1▲	VLAN ID
□	886	10.114.221.72	--	vxw-dvs-43-virtualwire-5-si...	Inactive	Link Do...	VLAN access: 154
□	893	10.114.221.73	--	vxw-dvs-43-virtualwire-5-si...	Inactive	Link Do...	VLAN access: 154
□	861	10.114.221.73	00:50:56:69:fd:99	vxw-vmknicPg-dvs-43-154...	--	Link Up	VLAN access: 154
□	863	10.114.221.72	00:50:56:86:07:96	vxw-dvs-43-virtualwire-2-si...	Inactive	Link Up	VLAN access: 154
□	869	10.114.221.72	00:50:56:86:51:cd	vxw-dvs-43-virtualwire-2-si...	Inactive	Link Up	VLAN access: 154
□	854	10.114.221.72	00:50:56:63:56:7b	vxw-vmknicPg-dvs-43-154...	--	Link Up	VLAN access: 154
□	876	10.114.221.73	00:50:56:86:00:9a	vxw-dvs-43-virtualwire-3-si...	Inactive	Link Up	VLAN access: 154
□	868	10.114.221.73	00:50:56:86:e2:a8	vxw-dvs-43-virtualwire-2-si...	Inactive	Link Up	VLAN access: 154
□	878	10.114.221.73	00:50:56:86:92:ab	vxw-dvs-43-virtualwire-4-si...	Inactive	Link Up	VLAN access: 154
□	381	10.114.221.73	00:50:56:69:e5:0f	Compute_DVS_Storage	--	Link Up	VLAN access: 155
□	380	10.114.221.72	00:50:56:6b:15:16	Compute_DVS_Storage	--	Link Up	VLAN access: 155
□	253	10.114.221.73	00:25:90:eb:ba:e8	Compute_DVS_Managem...	--	Link Up	VLAN access: 152

Click OK to save the selection.

[Note: Alternately, if you know the Port ID corresponding to the vNICs that must be mirrored, click the button to specify those source ports.]



Add Ports

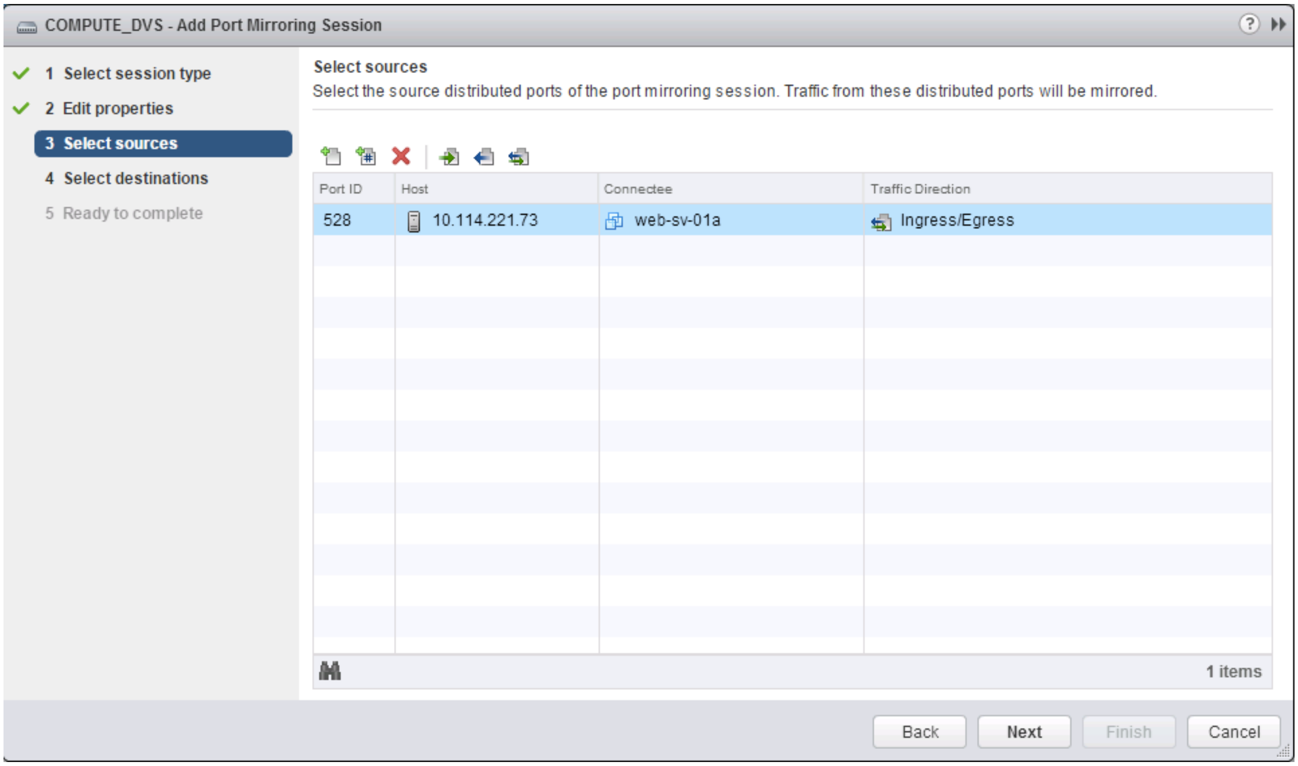
Port IDs(e.g. 1-4, 5, 10-21)

Step 8) Once you have selected the vNICs, select the direction of the traffic that must be mirrored. By default, both Ingress and Egress traffic will be mirrored on that vNIC. If you wish to only mirror Ingress or Egress, then use these buttons to toggle the configuration.

Note: Ingress = Traffic exiting the VM's vNIC and **Ingressing** the vSwitch port.

Egress = Traffic entering the VM's vNIC and **Egressing** the vSwitch port.

Click Next to specify the destination where the mirrored traffic is to be sent.



COMPUTE_DVS - Add Port Mirroring Session

1 Select session type
2 Edit properties
3 Select sources
4 Select destinations
5 Ready to complete

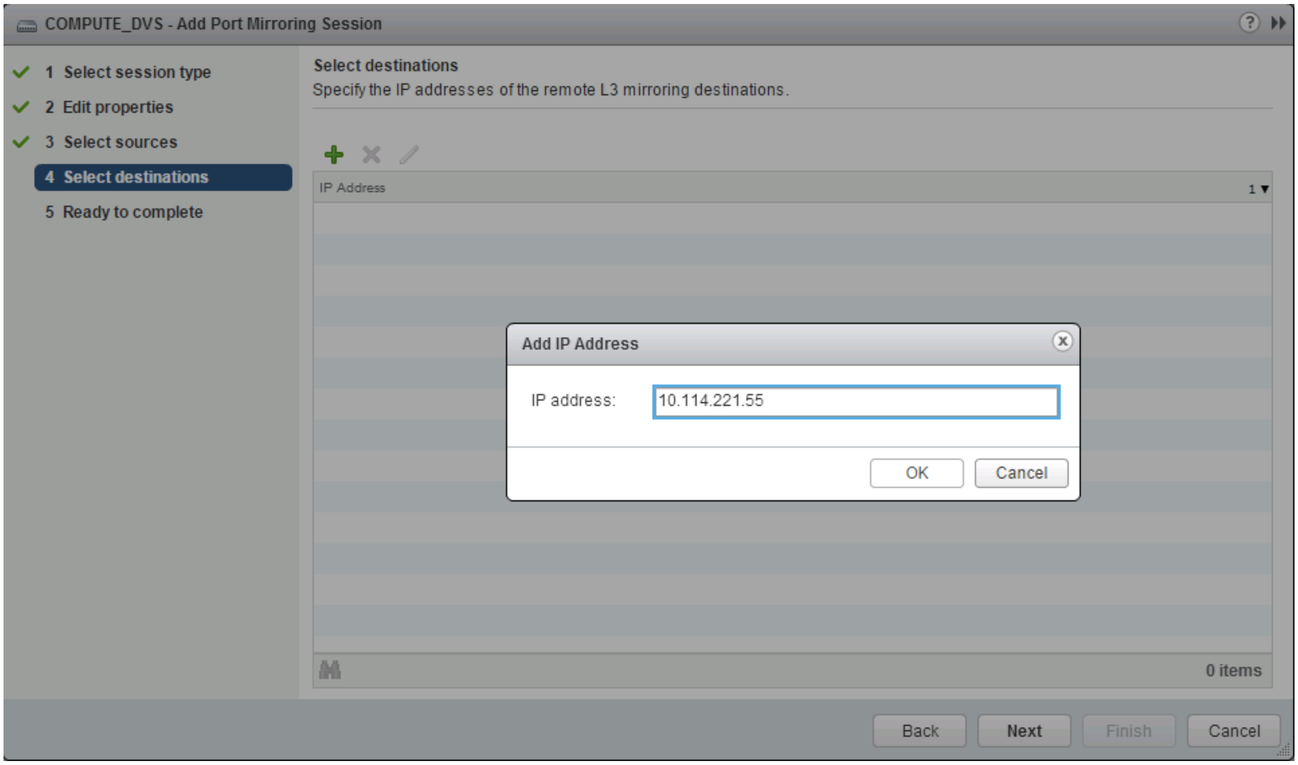
Select sources

Select the source distributed ports of the port mirroring session. Traffic from these distributed ports will be mirrored.

Port ID	Host	Connectee	Traffic Direction
528	10.114.221.73	web-sv-01a	Ingress/Egress

1 items

Step 8) Click (+) to add IP address as the destination for this mirrored traffic. Click OK and then click Next.



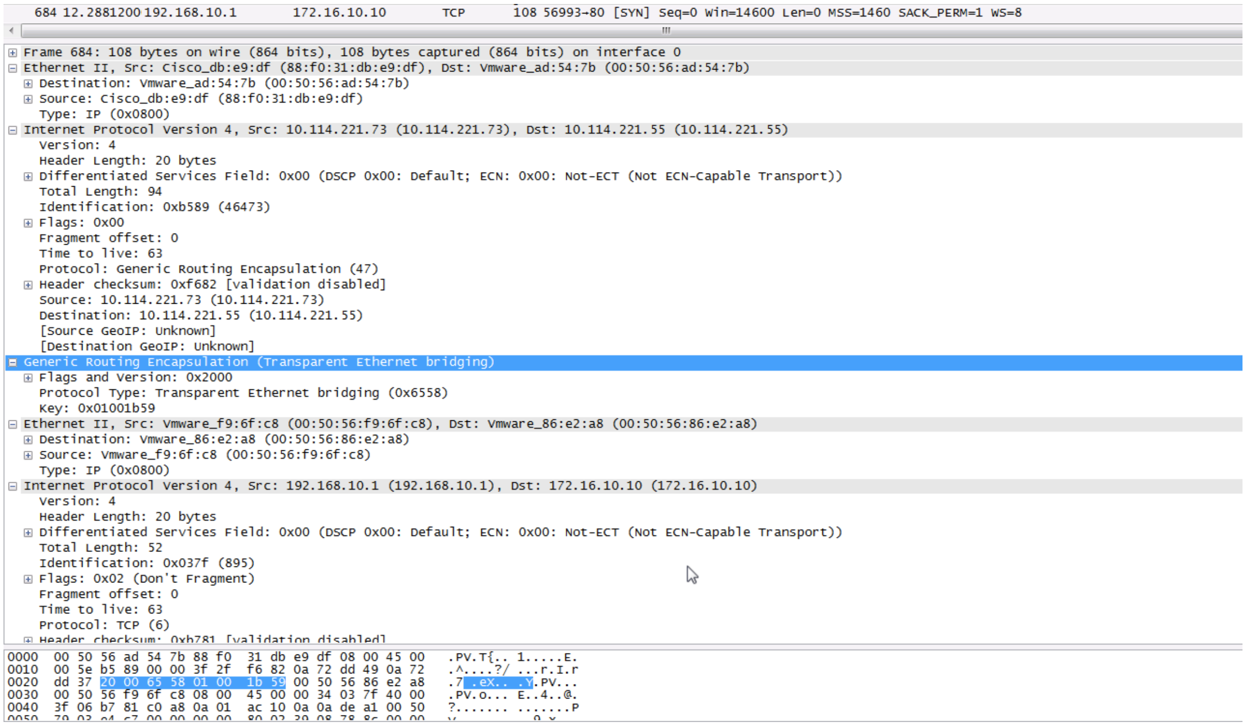
Select destinations
Specify the IP addresses of the remote L3 mirroring destinations.

Add IP Address

IP address: 10.114.221.55

Step 9) Click Finish to save the Port Mirroring session.

Here's screenshot of mirrored packets captured on Wireshark.



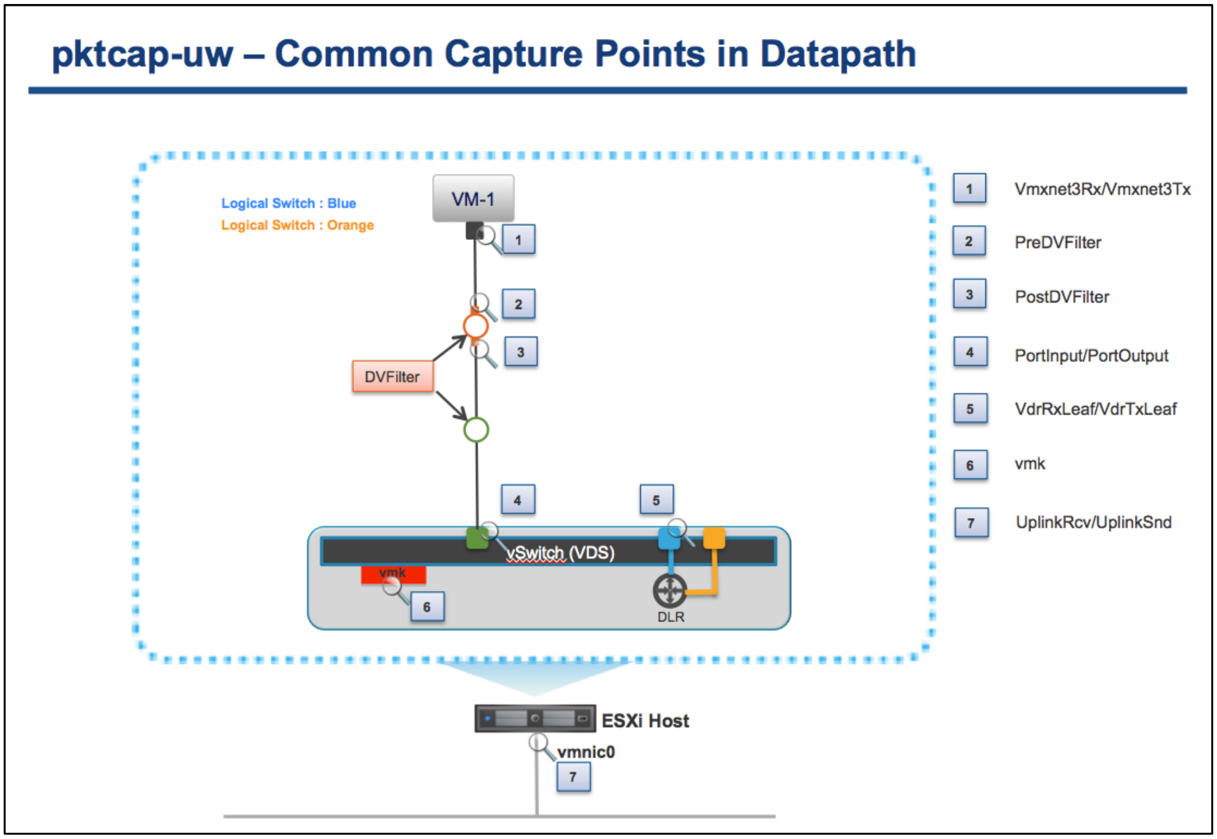
```
   684 12.2881200 192.168.10.1         172.16.10.10         TCP        108 56993→80 [SYN] Seq=0 Win=14600 Len=0 MSS=1460 SACK_PERM=1 WS=8

⊞ Frame 684: 108 bytes on wire (864 bits), 108 bytes captured (864 bits) on interface 0
⊟ Ethernet II, Src: Cisco_db:e9:df (88:f0:31:db:e9:df), Dst: Vmware_ad:54:7b (00:50:56:ad:54:7b)
  ⊞ Destination: Vmware_ad:54:7b (00:50:56:ad:54:7b)
  ⊞ Source: Cisco_db:e9:df (88:f0:31:db:e9:df)
    Type: IP (0x0800)
⊟ Internet Protocol Version 4, Src: 10.114.221.73 (10.114.221.73), Dst: 10.114.221.55 (10.114.221.55)
    Version: 4
    Header Length: 20 bytes
  ⊞ Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00: Not-ECT (Not ECN-Capable Transport))
    Total Length: 94
    Identification: 0xb589 (46473)
  ⊞ Flags: 0x00
    Fragment offset: 0
    Time to live: 63
    Protocol: Generic Routing Encapsulation (47)
  ⊞ Header checksum: 0xf682 [validation disabled]
    Source: 10.114.221.73 (10.114.221.73)
    Destination: 10.114.221.55 (10.114.221.55)
    [Source GeoIP: Unknown]
    [Destination GeoIP: Unknown]
⊟ Generic Routing Encapsulation (Transparent Ethernet bridging)
  ⊞ Flags and Version: 0x2000
    Protocol Type: Transparent Ethernet bridging (0x6558)
    Key: 0x01001b59
⊟ Ethernet II, Src: Vmware_f9:6f:c8 (00:50:56:f9:6f:c8), Dst: Vmware_86:e2:a8 (00:50:56:86:e2:a8)
  ⊞ Destination: Vmware_86:e2:a8 (00:50:56:86:e2:a8)
  ⊞ Source: Vmware_f9:6f:c8 (00:50:56:f9:6f:c8)
    Type: IP (0x0800)
⊟ Internet Protocol Version 4, Src: 192.168.10.1 (192.168.10.1), Dst: 172.16.10.10 (172.16.10.10)
    Version: 4
    Header Length: 20 bytes
  ⊞ Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00: Not-ECT (Not ECN-Capable Transport))
    Total Length: 52
    Identification: 0x037f (895)
  ⊞ Flags: 0x02 (Don't Fragment)
    Fragment offset: 0
    Time to live: 63
    Protocol: TCP (6)
  ⊞ Header checksum: 0xb781 [validation disabled]

0000  00 50 56 ad 54 7b 88 f0  31 db e9 df 08 00 45 00   .PV.T{.. 1.....E.
0010  00 5e b5 89 00 00 3f 2f  f6 82 0a 72 dd 49 0a 72   .^....?/ ...r.I.r
0020  dd 37 20 00 65 58 01 00  1b 59 00 50 56 86 e2 a8   .7 .eX.. .Y.PV...
0030  00 50 56 f9 6f c8 08 00  45 00 00 34 03 7f 40 00   .PV.o... E..4..@.
0040  3f 06 b7 81 c0 a8 0a 01  ac 10 0a 0a de a1 00 50   ?....... .......P
0050  79 03 e4 c7 00 00 00 00  80 02 39 08 78 8c 00 00   y....... ....9.x.
```

Host Level Packet Capture

Packet capture on ESXi host

vSphere 5.5 introduced an enhanced packet capture tool on the ESXi host. This tool allows traffic to be captured at all points within the hypervisor for greater flexibility and improved troubleshooting.

The common capture points are shown in the below picture.



pktcap-uw – Common Capture Points in Datapath

1	Vmxnet3Rx/Vmxnet3Tx
2	PreDVFilter
3	PostDVFilter
4	PortInput/PortOutput
5	VdrRxLeaf/VdrTxLeaf
6	vmk
7	UplinkRcv/UplinkSnd

Syntax

```
    pktcap-uw [--capture <capture point> | [--dir <0/1>] [--stage <0/1>]]
            [--switchport <PortID> | --vmk <vmknic> | --uplink <vmnic> |
              --dvfilter <filter name>] --lifID <lif id for vdr>]
            [-f [module name.]<function name>]
            [-AFhP] [-p|--port <Socket PORT>]
            [-c|--count <number>] [-s|--snapLen <length>]
            [-o|--outfile <FILE>] [--console] [Flow filter options]
```

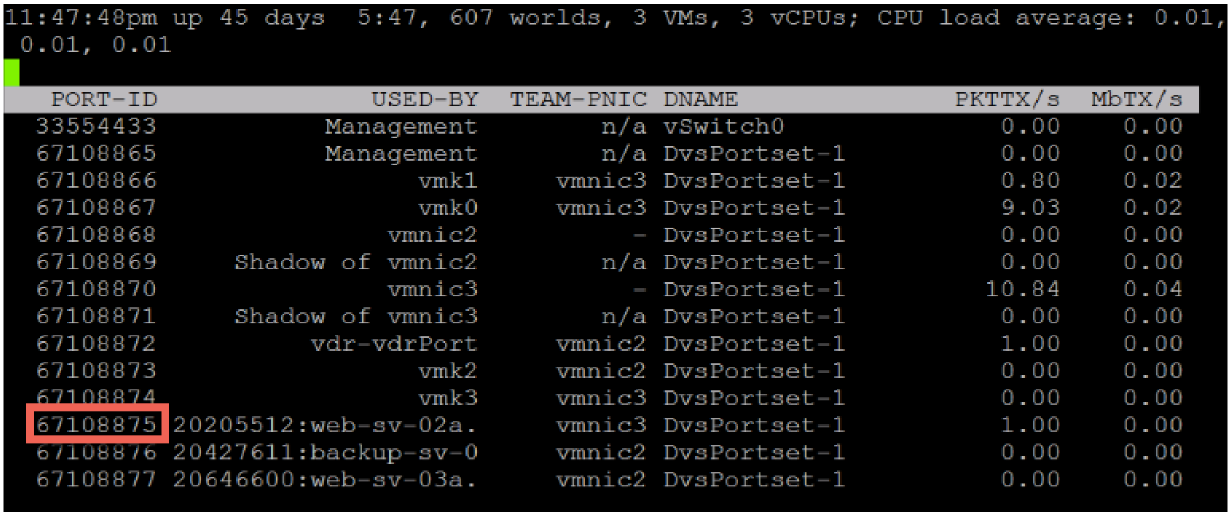
Example 1: To capture traffic at vNIC of a VM

Command: pktcap-uw --switchport *<vm port id>* --capture *Vmxnet3Rx*

Hint: Vmxnet3*Rx* will capture packet that is *received* by the Virtual Machine. To capture packet *transmitted* by the Virtual Machine, use --capture *Vmxnet3Tx*

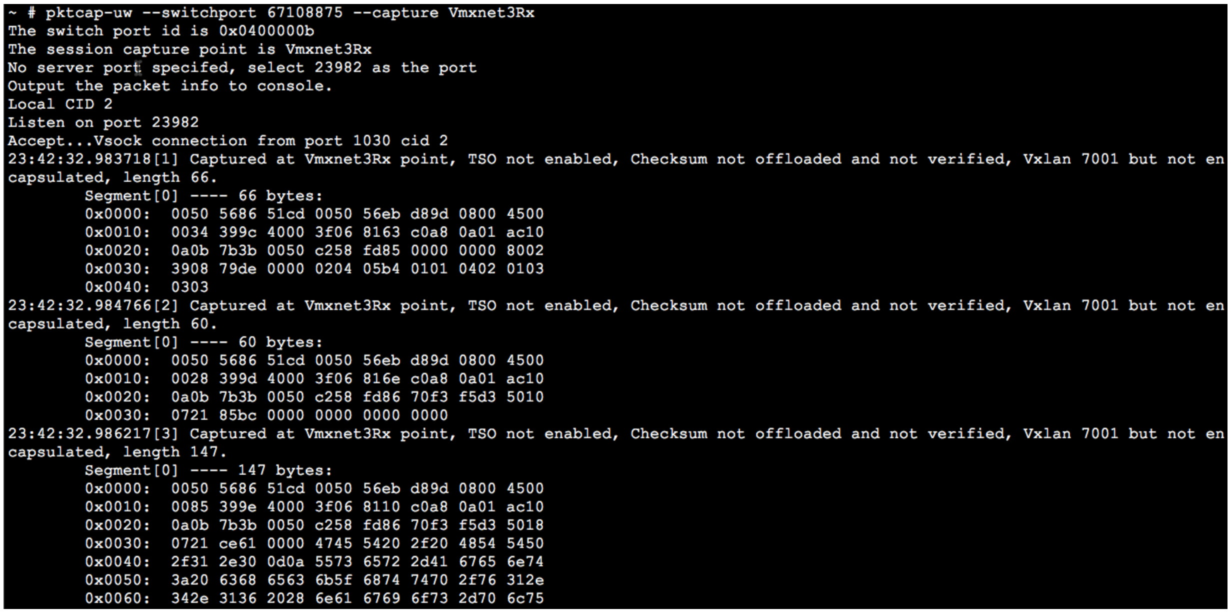
Note: The VM must use vmxnet3 NIC driver type to enable capturing packet at a VM's vNIC.

Step 1) Find the PORT-ID of the Virtual Machine. To do so, run *esxtop* command on the host and then pressing key *n*. In our example, lets capture traffic for web-sv-02a. The PORT-ID is 67108875 from the esxtop output.



```
11:47:48pm up 45 days  5:47, 607 worlds, 3 VMs, 3 vCPUs; CPU load average: 0.01,
 0.01, 0.01

   PORT-ID              USED-BY  TEAM-PNIC DNAME              PKTTX/s  MbTX/s
  33554433           Management        n/a vSwitch0              0.00    0.00
  67108865           Management        n/a DvsPortset-1          0.00    0.00
  67108866                 vmk1     vmnic3 DvsPortset-1          0.80    0.02
  67108867                 vmk0     vmnic3 DvsPortset-1          9.03    0.02
  67108868               vmnic2          - DvsPortset-1          0.00    0.00
  67108869     Shadow of vmnic2        n/a DvsPortset-1          0.00    0.00
  67108870               vmnic3          - DvsPortset-1         10.84    0.04
  67108871     Shadow of vmnic3        n/a DvsPortset-1          0.00    0.00
  67108872          vdr-vdrPort     vmnic2 DvsPortset-1          1.00    0.00
  67108873                 vmk2     vmnic2 DvsPortset-1          0.00    0.00
  67108874                 vmk3     vmnic3 DvsPortset-1          0.00    0.00
  67108875 20205512:web-sv-02a.     vmnic3 DvsPortset-1          1.00    0.00
  67108876 20427611:backup-sv-0     vmnic2 DvsPortset-1          0.00    0.00
  67108877 20646600:web-sv-03a.     vmnic2 DvsPortset-1          0.00    0.00
```

Step 2) Run the *pktcap-uw* command as shown in the below figure.



```
~ # pktcap-uw --switchport 67108875 --capture Vmxnet3Rx
The switch port id is 0x0400000b
The session capture point is Vmxnet3Rx
No server port specifed, select 23982 as the port
Output the packet info to console.
Local CID 2
Listen on port 23982
Accept...Vsock connection from port 1030 cid 2
23:42:32.983718[1] Captured at Vmxnet3Rx point, TSO not enabled, Checksum not offloaded and not verified, Vxlan 7001 but not en
capsulated, length 66.
        Segment[0] ---- 66 bytes:
        0x0000:  0050 5686 51cd 0050 56eb d89d 0800 4500
        0x0010:  0034 399c 4000 3f06 8163 c0a8 0a01 ac10
        0x0020:  0a0b 7b3b 0050 c258 fd85 0000 0000 8002
        0x0030:  3908 79de 0000 0204 05b4 0101 0402 0103
        0x0040:  0303
23:42:32.984766[2] Captured at Vmxnet3Rx point, TSO not enabled, Checksum not offloaded and not verified, Vxlan 7001 but not en
capsulated, length 60.
        Segment[0] ---- 60 bytes:
        0x0000:  0050 5686 51cd 0050 56eb d89d 0800 4500
        0x0010:  0028 399d 4000 3f06 816e c0a8 0a01 ac10
        0x0020:  0a0b 7b3b 0050 c258 fd86 70f3 f5d3 5010
        0x0030:  0721 85bc 0000 0000 0000 0000
23:42:32.986217[3] Captured at Vmxnet3Rx point, TSO not enabled, Checksum not offloaded and not verified, Vxlan 7001 but not en
capsulated, length 147.
        Segment[0] ---- 147 bytes:
        0x0000:  0050 5686 51cd 0050 56eb d89d 0800 4500
        0x0010:  0085 399e 4000 3f06 8110 c0a8 0a01 ac10
        0x0020:  0a0b 7b3b 0050 c258 fd86 70f3 f5d3 5018
        0x0030:  0721 ce61 0000 4745 5420 2f20 4854 5450
        0x0040:  2f31 2e30 0d0a 5573 6572 2d41 6765 6e74
        0x0050:  3a20 6368 6563 6b5f 6874 7470 2f76 312e
        0x0060:  342e 3136 2028 6e61 6769 6f73 2d70 6c75
```

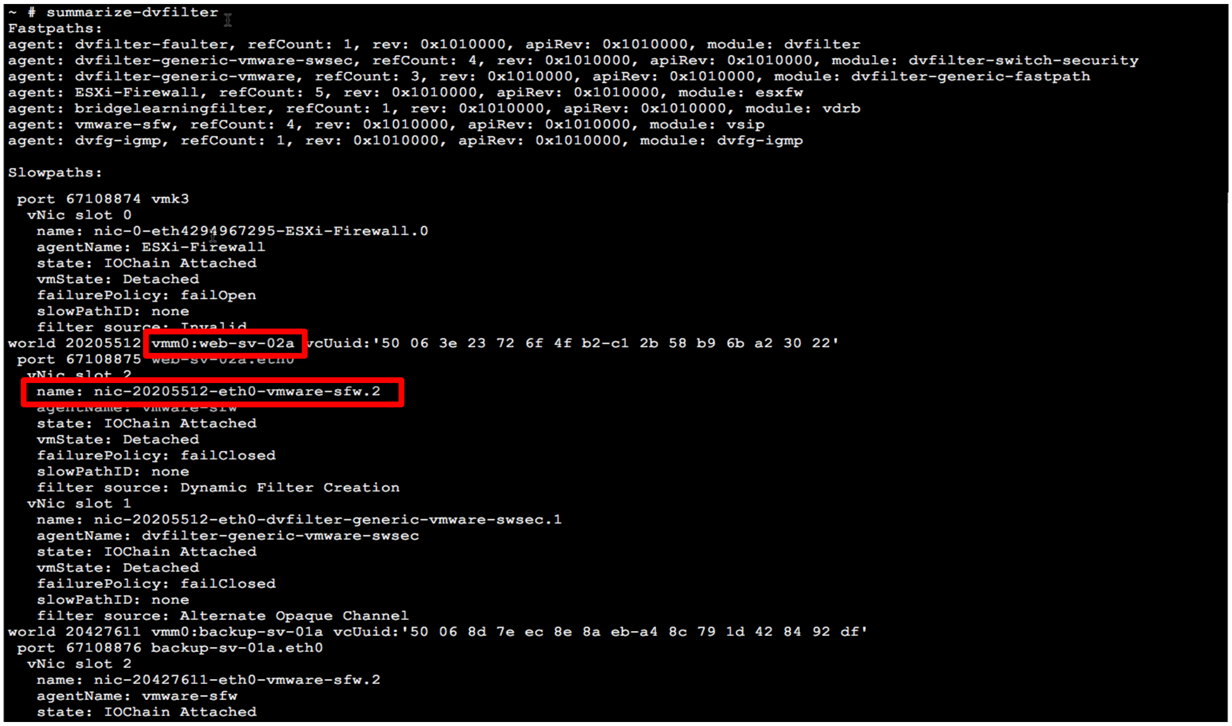
To save the output of the pktcap command in pcap format, use the option –o <outfile-file path/name>

Example 2: To capture packet before a DVFilter

Command:

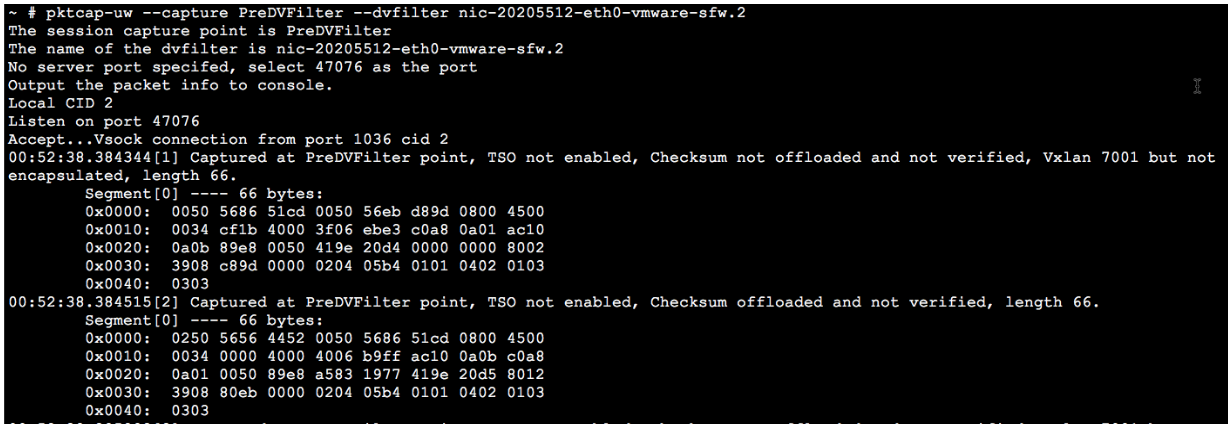
```
pktcap-uw --capture PreDVFilter --dvfilter <dvfilter-name>
```

Step 1: To find the dvfilter-name, run the command the *summarize-dvfilter* command on the host.



```
~ # summarize-dvfilter
Fastpaths:
agent: dvfilter-faulter, refCount: 1, rev: 0x1010000, apiRev: 0x1010000, module: dvfilter
agent: dvfilter-generic-vmware-swsec, refCount: 4, rev: 0x1010000, apiRev: 0x1010000, module: dvfilter-switch-security
agent: dvfilter-generic-vmware, refCount: 3, rev: 0x1010000, apiRev: 0x1010000, module: dvfilter-generic-fastpath
agent: ESXi-Firewall, refCount: 5, rev: 0x1010000, apiRev: 0x1010000, module: esxfw
agent: bridgelearningfilter, refCount: 1, rev: 0x1010000, apiRev: 0x1010000, module: vdrb
agent: vmware-sfw, refCount: 4, rev: 0x1010000, apiRev: 0x1010000, module: vsip
agent: dvfg-igmp, refCount: 1, rev: 0x1010000, apiRev: 0x1010000, module: dvfg-igmp

Slowpaths:

 port 67108874 vmk3
  vNic slot 0
   name: nic-0-eth4294967295-ESXi-Firewall.0
   agentName: ESXi-Firewall
   state: IOChain Attached
   vmState: Detached
   failurePolicy: failOpen
   slowPathID: none
   filter source: Invalid
world 20205512 vmm0:web-sv-02a vcUuid:'50 06 3e 23 72 6f 4f b2-c1 2b 58 b9 6b a2 30 22'
 port 67108875 web-sv-02a.eth0
  vNic slot 2
   name: nic-20205512-eth0-vmware-sfw.2
   agentName: vmware-sfw
   state: IOChain Attached
   vmState: Detached
   failurePolicy: failClosed
   slowPathID: none
   filter source: Dynamic Filter Creation
  vNic slot 1
   name: nic-20205512-eth0-dvfilter-generic-vmware-swsec.1
   agentName: dvfilter-generic-vmware-swsec
   state: IOChain Attached
   vmState: Detached
   failurePolicy: failClosed
   slowPathID: none
   filter source: Alternate Opaque Channel
world 20427611 vmm0:backup-sv-01a vcUuid:'50 06 8d 7e ec 8e 8a eb-a4 8c 79 1d 42 84 92 df'
 port 67108876 backup-sv-01a.eth0
  vNic slot 2
   name: nic-20427611-eth0-vmware-sfw.2
   agentName: vmware-sfw
   state: IOChain Attached
```

Step 2 : Run pktcap-uw command as shown below



```
~ # pktcap-uw --capture PreDVFilter --dvfilter nic-20205512-eth0-vmware-sfw.2
The session capture point is PreDVFilter
The name of the dvfilter is nic-20205512-eth0-vmware-sfw.2
No server port specifed, select 47076 as the port
Output the packet info to console.
Local CID 2
Listen on port 47076
Accept...Vsock connection from port 1036 cid 2
00:52:38.384344[1] Captured at PreDVFilter point, TSO not enabled, Checksum not offloaded and not verified, Vxlan 7001 but not
encapsulated, length 66.
        Segment[0] ---- 66 bytes:
        0x0000:  0050 5686 51cd 0050 56eb d89d 0800 4500
        0x0010:  0034 cf1b 4000 3f06 ebe3 c0a8 0a01 ac10
        0x0020:  0a0b 89e8 0050 419e 20d4 0000 0000 8002
        0x0030:  3908 c89d 0000 0204 05b4 0101 0402 0103
        0x0040:  0303
00:52:38.384515[2] Captured at PreDVFilter point, TSO not enabled, Checksum offloaded and not verified, length 66.
        Segment[0] ---- 66 bytes:
        0x0000:  0250 5656 4452 0050 5686 51cd 0800 4500
        0x0010:  0034 0000 4000 4006 b9ff ac10 0a0b c0a8
        0x0020:  0a01 0050 89e8 a583 1977 419e 20d5 8012
        0x0030:  3908 80eb 0000 0204 05b4 0101 0402 0103
        0x0040:  0303
```

To capture packet after the DVFilter processing/redirection, then use **`--capture`** ***`PostDVFilter`***

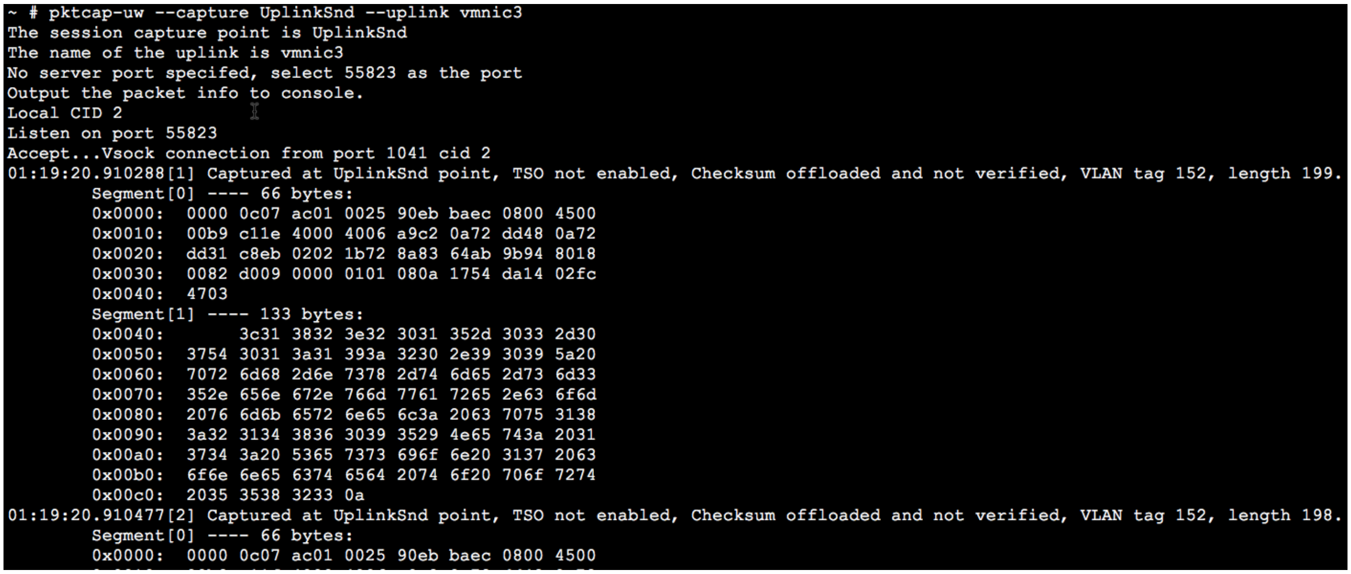
Example 3: To capture packet on a uplink on the host

Command:

```
pktcap-uw --capture UplinkSnd | UplinkRcv --uplink <vmnic-name>
```

UplinkSnd - Packets send on the uplink

UplinkRcv - Packets received on the uplink



```
~ # pktcap-uw --capture UplinkSnd --uplink vmnic3
The session capture point is UplinkSnd
The name of the uplink is vmnic3
No server port specifed, select 55823 as the port
Output the packet info to console.
Local CID 2
Listen on port 55823
Accept...Vsock connection from port 1041 cid 2
01:19:20.910288[1] Captured at UplinkSnd point, TSO not enabled, Checksum offloaded and not verified, VLAN tag 152, length 199.
        Segment[0] ---- 66 bytes:
        0x0000:  0000 0c07 ac01 0025 90eb baec 0800 4500
        0x0010:  00b9 c11e 4000 4006 a9c2 0a72 dd48 0a72
        0x0020:  dd31 c8eb 0202 1b72 8a83 64ab 9b94 8018
        0x0030:  0082 d009 0000 0101 080a 1754 da14 02fc
        0x0040:  4703
        Segment[1] ---- 133 bytes:
        0x0040:       3c31 3832 3e32 3031 352d 3033 2d30
        0x0050:  3754 3031 3a31 393a 3230 2e39 3039 5a20
        0x0060:  7072 6d68 2d6e 7378 2d74 6d65 2d73 6d33
        0x0070:  352e 656e 672e 766d 7761 7265 2e63 6f6d
        0x0080:  2076 6d6b 6572 6e65 6c3a 2063 7075 3138
        0x0090:  3a32 3134 3836 3039 3529 4e65 743a 2031
        0x00a0:  3734 3a20 5365 7373 696f 6e20 3137 2063
        0x00b0:  6f6e 6e65 6374 6564 2074 6f20 706f 7274
        0x00c0:  2035 3538 3233 0a
01:19:20.910477[2] Captured at UplinkSnd point, TSO not enabled, Checksum offloaded and not verified, VLAN tag 152, length 198.
        Segment[0] ---- 66 bytes:
        0x0000:  0000 0c07 ac01 0025 90eb baec 0800 4500
```

Other Examples

To capture packet sent by vmkernel port

Command:

```
pktcap-uw --vmk <vmknic>
```

To capture packet on a vSwitch port

Command:

```
pktcap-uw --switchport <port id> --dir 0 --stage 1
```

In this case, since we do not use --capture option, we can use the --dir options to specify the direction (0=in and 1=out) and --stage option to denote pre- (0) and post- (1) stage to capture traffic

Note: In previous examples, where we used --capture option, we were implicitly specifying the direction and stage of traffic, based on the argument that was supplied. e.g. Vmxnet3Tx, PreDVFilter or th

To capture packet received by vmkernel port, use the --dir 1 parameter.

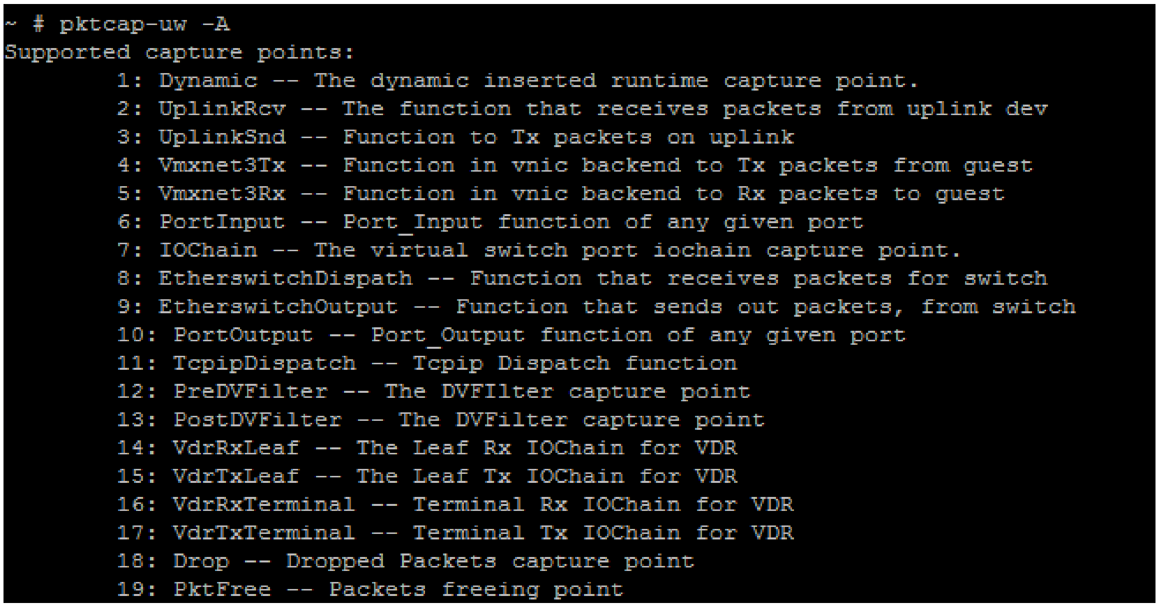
```
pktcap-uw --vmk <vmknic> --dir 1
```

To capture all the dropped packets on a host

```
pktcap-uw --capture Drop
```

To list all available capture points, type

```
pktcap-uw –A
```



```
~ # pktcap-uw -A
Supported capture points:
        1: Dynamic -- The dynamic inserted runtime capture point.
        2: UplinkRcv -- The function that receives packets from uplink dev
        3: UplinkSnd -- Function to Tx packets on uplink
        4: Vmxnet3Tx -- Function in vnic backend to Tx packets from guest
        5: Vmxnet3Rx -- Function in vnic backend to Rx packets to guest
        6: PortInput -- Port_Input function of any given port
        7: IOChain -- The virtual switch port iochain capture point.
        8: EtherswitchDispath -- Function that receives packets for switch
        9: EtherswitchOutput -- Function that sends out packets, from switch
        10: PortOutput -- Port_Output function of any given port
        11: TcpipDispatch -- Tcpip Dispatch function
        12: PreDVFilter -- The DVFIlter capture point
        13: PostDVFilter -- The DVFilter capture point
        14: VdrRxLeaf -- The Leaf Rx IOChain for VDR
        15: VdrTxLeaf -- The Leaf Tx IOChain for VDR
        16: VdrRxTerminal -- Terminal Rx IOChain for VDR
        17: VdrTxTerminal -- Terminal Tx IOChain for VDR
        18: Drop -- Dropped Packets capture point
        19: PktFree -- Packets freeing point
```

Common Traffic capture filters

By default, the above commands would capture all the traffic on a specified capture point. Flow filter options can be specified with the above commands to capture the traffic that matches the filter criteria.

Flow filter options, it will be applied when set:

--srcmac <xx:xx:xx:xx:xx>

The Ethernet source MAC address.

--dstmac <xx:xx:xx:xx:xx>

The Ethernet destination MAC address.

--mac <xx:xx:xx:xx:xx>

The Ethernet MAC address(src or dst).

--ethtype 0x<ETHTYPE>

The Ethernet type. HEX format.

--vlan <VLANID>

The Ethernet VLAN ID.

--srcip <x.x.x.x[/<range>]>

The source IP address.

--dstip <x.x.x.x[/<range>]>

The destination IP address.

--ip <x.x.x.x>

The IP address(src or dst).

--proto 0x<IPPROTYPE>

The IP protocol.

--srcport <SRCPORT>

The TCP source port.

--dstport <DSTPORT>

The TCP destination port.

--tcpport <PORT>

The TCP port(src or dst).

--vxlan <vxlan id>

The vxlan id of flow.

Common Traffic capture options:

```
-o|--outfile <FILENAME> : To save output as a pcap file.
-s|--snaplen <length>   : To capture first <length> packet buffer
-c|--count <NUMBER>     : NUMBER of packets to capture
```

Packet capture on NSX Edge VM

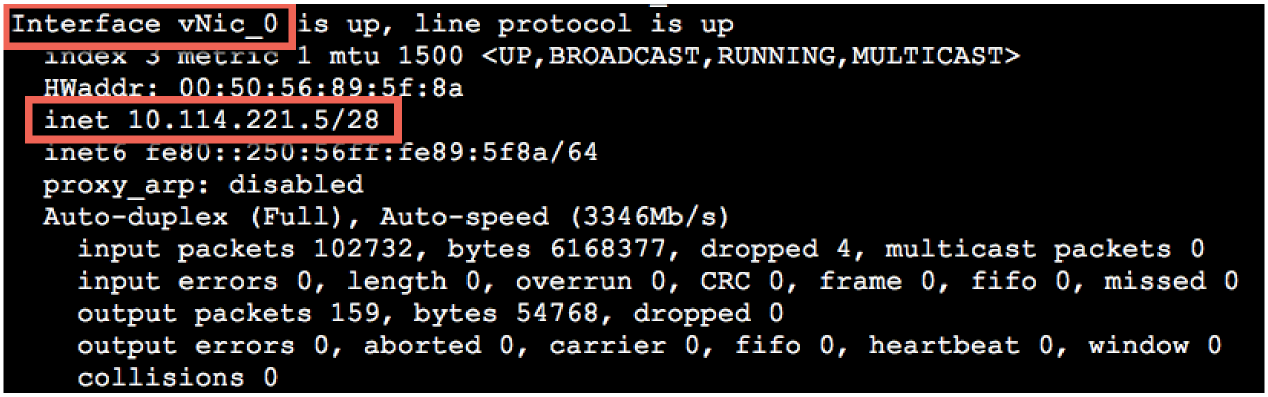
NSX Edge provides capturing of traffic on a vNIC (Internal or Uplink). The captured traffic can be display on the terminal or saved on the Edge VM locally as a file.

Example: Capture packet on ESG's interface connected to VLAN Port-group (Uplink to Physical Next-hop).

Command:

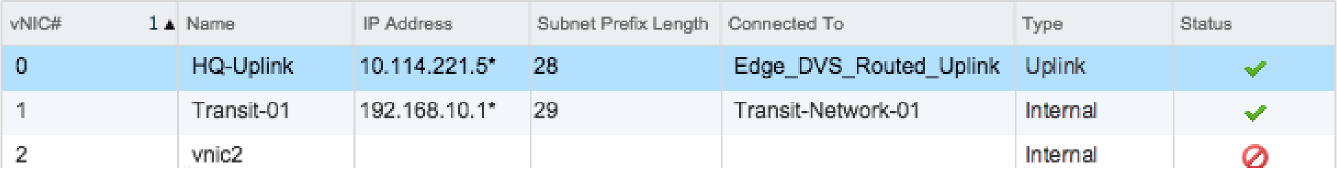
```
debug packet [display|capture]
```

Run "show interface" to get the interface ID. In my example, I would like to capture traffic on the vNic_0, which is the uplink. This is verified by the IP address assigned to this interface.



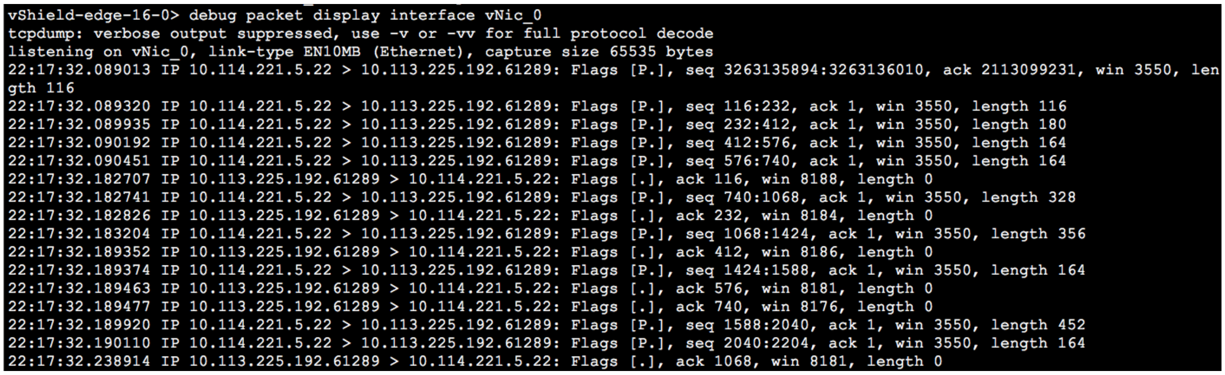
```
Interface vNic_0 is up, line protocol is up
  index 3 metric 1 mtu 1500 <UP,BROADCAST,RUNNING,MULTICAST>
  HWaddr: 00:50:56:89:5f:8a
  inet 10.114.221.5/28
  inet6 fe80::250:56ff:fe89:5f8a/64
  proxy_arp: disabled
  Auto-duplex (Full), Auto-speed (3346Mb/s)
    input packets 102732, bytes 6168377, dropped 4, multicast packets 0
    input errors 0, length 0, overrun 0, CRC 0, frame 0, fifo 0, missed 0
    output packets 159, bytes 54768, dropped 0
    output errors 0, aborted 0, carrier 0, fifo 0, heartbeat 0, window 0
    collisions 0
```

Alternately, this information can be found from the configuration of the NSX Edge on the NSX Manager.



vNIC# 1 ▲	Name	IP Address	Subnet Prefix Length	Connected To	Type	Status
0	HQ-Uplink	10.114.221.5*	28	Edge_DVS_Routed_Uplink	Uplink	✓
1	Transit-01	192.168.10.1*	29	Transit-Network-01	Internal	✓
2	vnic2				Internal	⊘

Enter the command to capture



```
vShield-edge-16-0> debug packet display interface vNic_0
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on vNic_0, link-type EN10MB (Ethernet), capture size 65535 bytes
22:17:32.089013 IP 10.114.221.5.22 > 10.113.225.192.61289: Flags [P.], seq 3263135894:3263136010, ack 2113099231, win 3550, len
gth 116
22:17:32.089320 IP 10.114.221.5.22 > 10.113.225.192.61289: Flags [P.], seq 116:232, ack 1, win 3550, length 116
22:17:32.089935 IP 10.114.221.5.22 > 10.113.225.192.61289: Flags [P.], seq 232:412, ack 1, win 3550, length 180
22:17:32.090192 IP 10.114.221.5.22 > 10.113.225.192.61289: Flags [P.], seq 412:576, ack 1, win 3550, length 164
22:17:32.090451 IP 10.114.221.5.22 > 10.113.225.192.61289: Flags [P.], seq 576:740, ack 1, win 3550, length 164
22:17:32.182707 IP 10.113.225.192.61289 > 10.114.221.5.22: Flags [.], ack 116, win 8188, length 0
22:17:32.182741 IP 10.114.221.5.22 > 10.113.225.192.61289: Flags [P.], seq 740:1068, ack 1, win 3550, length 328
22:17:32.182826 IP 10.113.225.192.61289 > 10.114.221.5.22: Flags [.], ack 232, win 8184, length 0
22:17:32.183204 IP 10.114.221.5.22 > 10.113.225.192.61289: Flags [P.], seq 1068:1424, ack 1, win 3550, length 356
22:17:32.189352 IP 10.113.225.192.61289 > 10.114.221.5.22: Flags [.], ack 412, win 8186, length 0
22:17:32.189374 IP 10.114.221.5.22 > 10.113.225.192.61289: Flags [P.], seq 1424:1588, ack 1, win 3550, length 164
22:17:32.189463 IP 10.113.225.192.61289 > 10.114.221.5.22: Flags [.], ack 576, win 8181, length 0
22:17:32.189477 IP 10.113.225.192.61289 > 10.114.221.5.22: Flags [.], ack 740, win 8176, length 0
22:17:32.189920 IP 10.114.221.5.22 > 10.113.225.192.61289: Flags [P.], seq 1588:2040, ack 1, win 3550, length 452
22:17:32.190110 IP 10.114.221.5.22 > 10.113.225.192.61289: Flags [P.], seq 2040:2204, ack 1, win 3550, length 164
22:17:32.238914 IP 10.113.225.192.61289 > 10.114.221.5.22: Flags [.], ack 1068, win 8181, length 0
```

System events and status (via API)

This section demonstrates how to retrieve NSX system events from the NSX REST API. This section also provides suggestions for parsing the API output.

The VMware NSX REST API allows you to automate monitor events related to NSX and NSX-managed logical networks. REST API calls can be invoked using web browser REST CLIENT plug-in (for example, DHC) for browsers like Firefox and Chrome or using the cURL command line REST client. In these examples, we use DHC and cURL. See the appendix, "Preparing to use the NSX REST API," for instructions on setting up your REST client to use the NSX REST API.

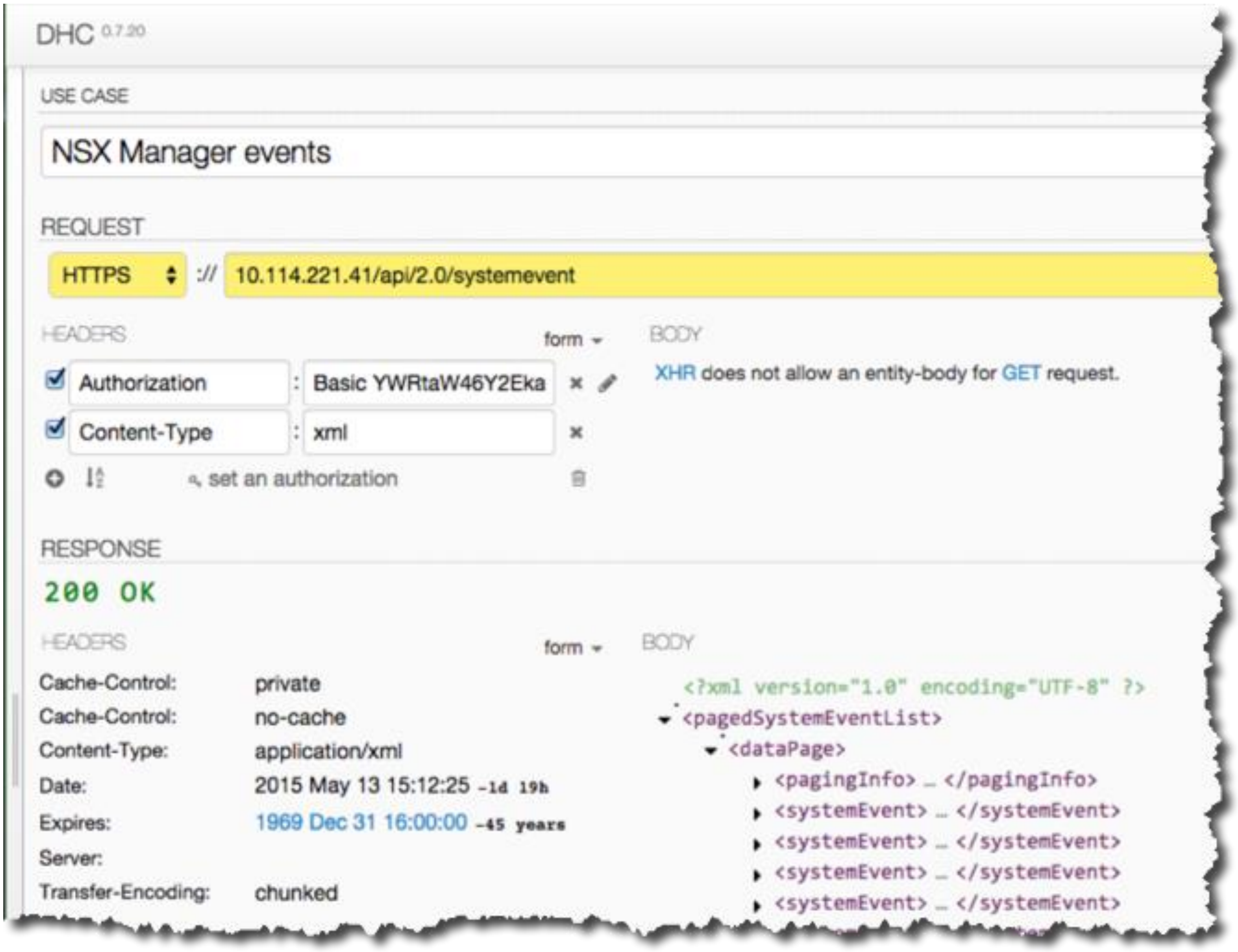
In the examples below, 10.114.221.41 is the IP Address of NSX Manager.

Note: See also the later section, "Logging" for information on gathering event information in NSX log files.

Viewing NSX system events using an in-browser REST client

Given below is a screenshot in which we use the Chrome web browser is used with DHC REST/HTTP API client to make the "systemevent" GET call to the API on NSX Manager. The API call is

```
GET https://10.114.221.41/api/2.0/systemevent
```



NSX Manager events

HTTPS :// 10.114.221.41/api/2.0/systemevent

☑ Authorization : Basic YWRtaW46Y2Eka

☑ Content-Type : xml

XHR does not allow an entity-body for GET request.

200 OK

Cache-Control:	private
Cache-Control:	no-cache
Content-Type:	application/xml
Date:	2015 May 13 15:12:25 -1d 19h
Expires:	1969 Dec 31 16:00:00 -45 years
Server:	
Transfer-Encoding:	chunked

```
<?xml version="1.0" encoding="UTF-8" ?>
<pagedSystemEventList>
  <dataPage>
    <pagingInfo> … </pagingInfo>
    <systemEvent> … </systemEvent>
    <systemEvent> … </systemEvent>
    <systemEvent> … </systemEvent>
    <systemEvent> … </systemEvent>
```

Multiple System events are recorded in the Response Window above. You can expand the <systemEvent> tab to see the details of a specific systemEvent. Below we show a sample segment of the output from the above command.

```
<systemEvent>
    <eventId> 29 </eventId>
    <timestamp> 1410828.123495 </timestamp>
    <severity> Critical </severity>
    <eventsource> Host messaging infrastructure </eventsource>
    <eventCode> 391002 </eventCode>
    <message> Messaging infrastructure down on host. </message>
    <module> Messaging infrastructure. </module>
    <objectId> host-28 </objectId>
    <reporterName> NSX Manager </reporterName>
    <reporterType> 1 </reporterType>
    <sourceType> 1 </sourceType>
    <eventMetadata/>
</systemEvent>
```

Viewing NSX system events using the cURL REST client

Here is an example of using the NSX REST API from the command line using cURL to capture all the system events:

```
# curl –insecure https://admin:admin@10.114.221.41/api/2.0/systemevent
```

For more information, see the *NSX for vSphere API Guide*, in the section, "Querying NSX Manager Logs."

Example: Show NSX Edge appliance status

This example shows how to use the NSX REST API from the command line using curl to show the NSX Edge appliance status

```
#curl --insecure https://admin:admin@10.114.221.41/api/4.0/edges/edge-16/status
```

Below we show a sample segment of the output from the above command:

```
<?xml version="1.0" encoding="UTF-8"?><edgeStatus><timestamp>1431715832689</timestamp>
    <systemStatus>good</systemStatus>
    <activeVseHaIndex>0</activeVseHaIndex>
    <edgeStatus>GREEN</edgeStatus>
    <publishStatus>APPLIED</publishStatus>
```

```
<version>84</version>

<edgeVmStatus/>

    <featureStatuses>

        <featureStatus>

            <service>dns</service>

            <status>up</status>

        </featureStatus>

        <featureStatus>

            <service>dhcp</service>

            <status>down</status>

        </featureStatus>

        <featureStatus>

            <service>highAvailability</service>

            <status>up</status>

        </featureStatus>

        <featureStatus>

            <service>loadBalancer</service>

            <status>up</status>

        </featureStatus>

        <featureStatus>

            <service>nat</service>

            <status>Applied</status>

        </featureStatus>

        <featureStatus>

            <service>syslog</service>

            <status>down</status>

        </featureStatus>

        <featureStatus>

            <service>firewall</service>

            <status>Applied</status>

        </featureStatus>

        <featureStatus>

            <service>sslvpn</service>

            <status>not_configured</status>

        </featureStatus>
        <featureStatus>
            <service>l2vpn</service>
            <status>not_configured</status>
        </featureStatus>
        <featureStatus>
            <service>routing</service>
            <status>Applied</status>
        </featureStatus>
        <featureStatus>
            <service>ipsec</service>
            <status>not_configured</status>
        </featureStatus>
    <featureStatuses>
```

For more information, see the *NSX for vSphere API Guide*, in the section, "Query Edge Status."

You can use curl commands in a script to periodically collect the responses in a file. An example of such a script is shown below where the API call collects the Edge status periodically and the response of the API call is send to a file named "edgestatus.txt."

```
#!/bin/bash
while true; do
curl --insecure https://admin:admin@10.114.221.41/api/4.0/edges/edge-16/status --o
edgestatus.txt
echo ""
sleep 60
done
```

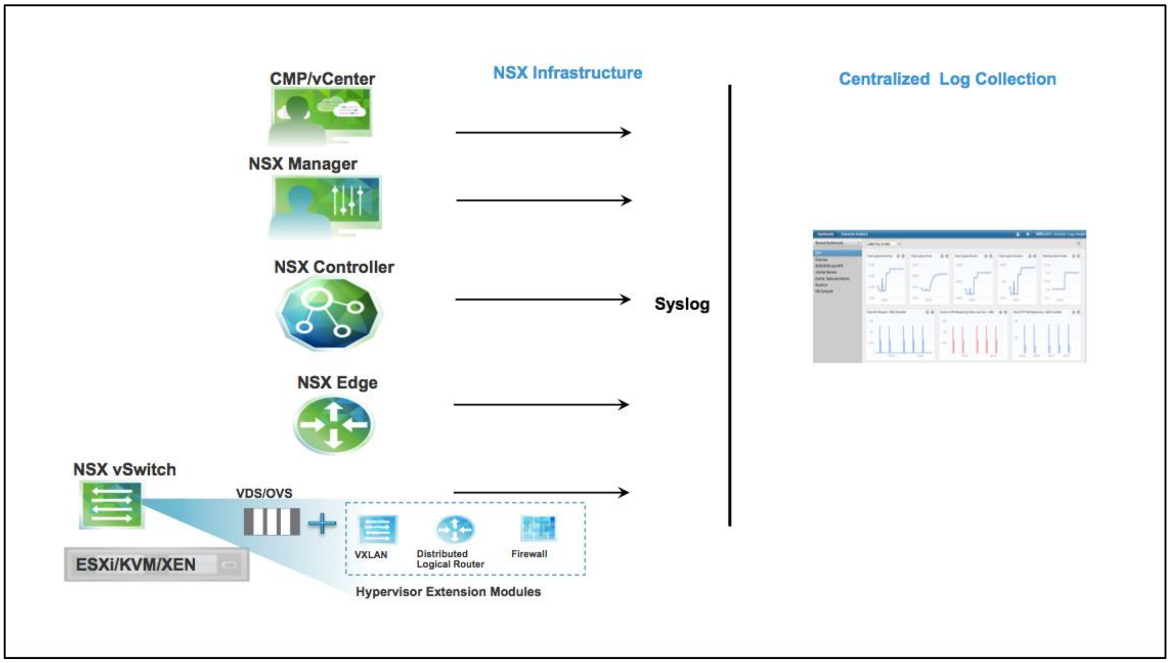
You can use a script to parse through the response file for specific data and use that data for monitoring purposes. For example, you can run the above script and within the response, filter for “><edgeStatus>GREEN” to verify the status of the Edge Appliance.

Alternatively, you can write a python script to run the API command periodically and define actions within the script for certain object values.

For more details on NSX REST APIs available for operations and monitoring, see the *NSX for vSphere API Guide*.

Logging

System Event Logs can be collected on vCenter, NSX Manager, NSX Controller, ESXi hosts and NSX Edge appliances. (See also the earlier section, " System events and status (via API)" for information on gathering event information via the API.)



Reference Table

The following reference table provides information on logs that are of importance to debugging various NSX Services. Refer to individual sections below for additional details and explanation on the log files.

Logs From Debugging	NSX Manager	NSX Controller	Hypervisor	Edge VM
Logical Switch	✓	✓	✓	
Logical Router	✓	✓	✓	✓
Distributed Firewall	✓		✓	
Edge Services	✓			✓

Log Rotation Schedule

NSX appliances retain log information according to the following log rotation schedule. The log rotation schedule is automatic and cannot be changed. The retention schedule is:

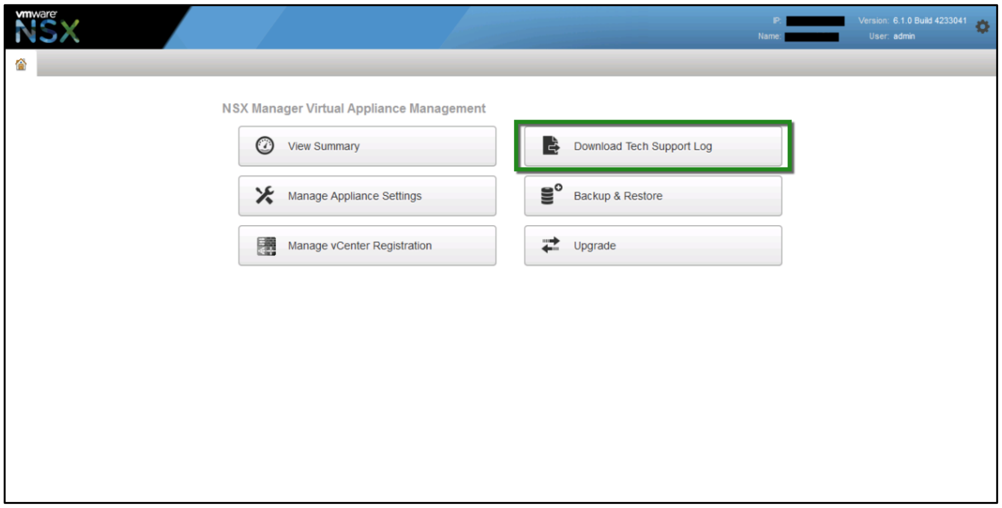
- NSX Manager: vsm.log rotated after 200MB, max 10 files are retained. Files are compressed when stored.
- NSX Controller: Log files rotated after 100MB, max 5 files are retained. Files are compressed when stored.
- NSX Edge: All logs are stored in /var/log/messages, rotated after 2MB, with max of 5 files retained.

VMware recommends using syslog for long term retention of logs. All NSX components support syslog. Since the disk space on each appliance (NSX Manager, NSX Controller and NSX Edge) is limited by the VM size, the log rotation policy is primarily based on size (and not based on time). For syslog set-up instructions see the section, "System Events and Audit Logs" in the *NSX Administrator Guide*.

NSX Manager Logs

NSX Manager - Tech Support Logs

NSX Manager reports SystemEvent for Logical Switch, Logical Router, Distributed Firewall and Edge Services. These logs can be access by downloading the Tech Support Log bundle. Tech Support logs can be manually downloaded from the NSX Manager Administration GUI at Home → Download Tech Support Log. This will generate a gzip file that can be downloaded for viewing/troubleshooting.



IP: [redacted]
Name: [redacted]
Version: 6.1.0 Build 4233041
User: admin

vmware NSX
NSX Manager Virtual Appliance Management
View Summary
Download Tech Support Log
Manage Appliance Settings
Backup & Restore
Manage vCenter Registration
Upgrade

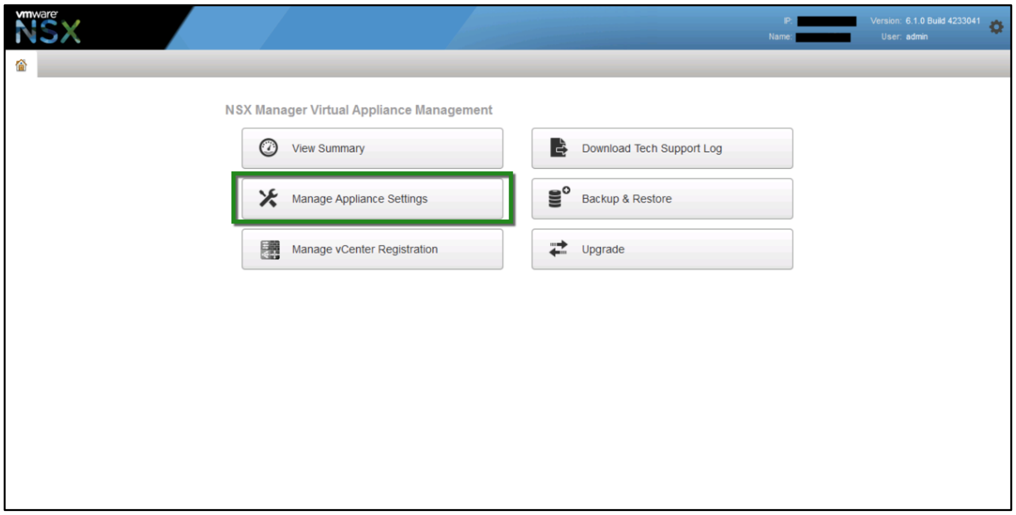
NSX Manager - Remote Syslog Server

NSX Manager can be configured to send Audit logs and System Events to a remote syslog server such as VMware vCenter Log Insight for Analytics or any other standard syslog servers.

Enabling Syslog Server on NSX Manager:

Step 1) Login to NSX Manager.

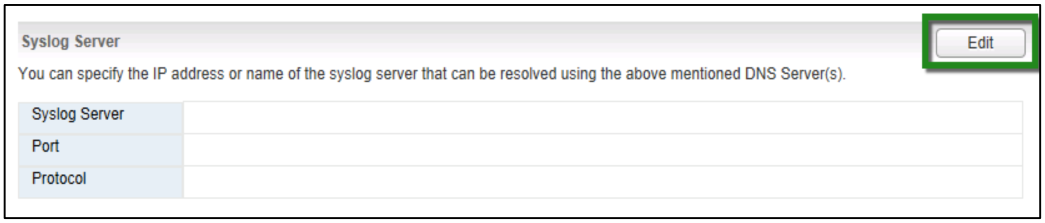
Step 2) Click on Manage Appliance Settings



IP: [redacted]
Name: [redacted]
Version: 6.1.0 Build 4233041
User: admin

vmware NSX
NSX Manager Virtual Appliance Management
View Summary
Download Tech Support Log
Manage Appliance Settings
Backup & Restore
Manage vCenter Registration
Upgrade

Step 3) Click on Edit button for Syslog Server as shown in figure below

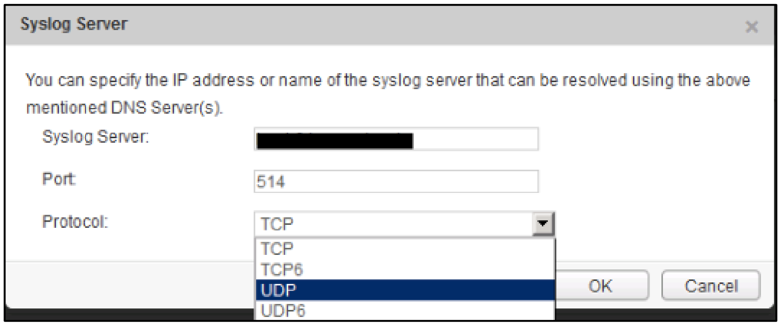


Syslog Server

You can specify the IP address or name of the syslog server that can be resolved using the above mentioned DNS Server(s).

Syslog Server	
Port	
Protocol	

Step 4) Enter the IP/FQDN of Syslog Server, Port (default port is 514) and protocol (UDP/UDP6/TCP/TCP6). If you are using FQDN, please ensure that DNS server is configured in the section above the Syslog Server.



Syslog Server

You can specify the IP address or name of the syslog server that can be resolved using the above mentioned DNS Server(s).

Syslog Server: [redacted]

Port: 514

Protocol: TCP

Step 5) Click OK

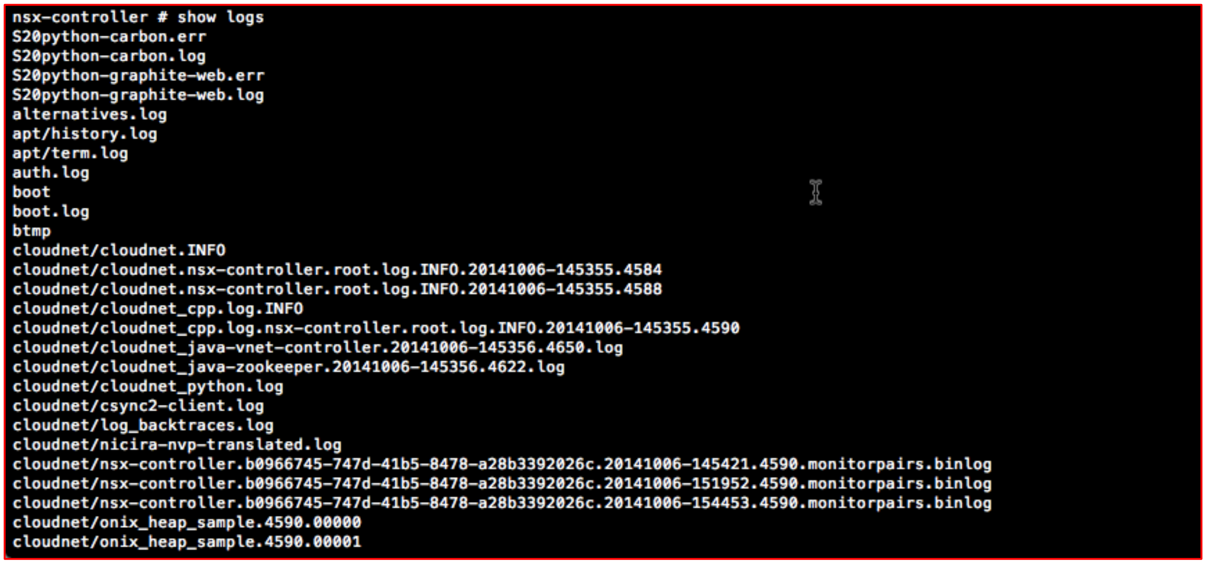
NSX Controller Logs

NSX Controller Logs via CLI

NSX controller is an advanced distributed state management system that controls virtual networks and overlay transport tunnels.

The following commands can be used to access log files on the NSX Controller:

show logs

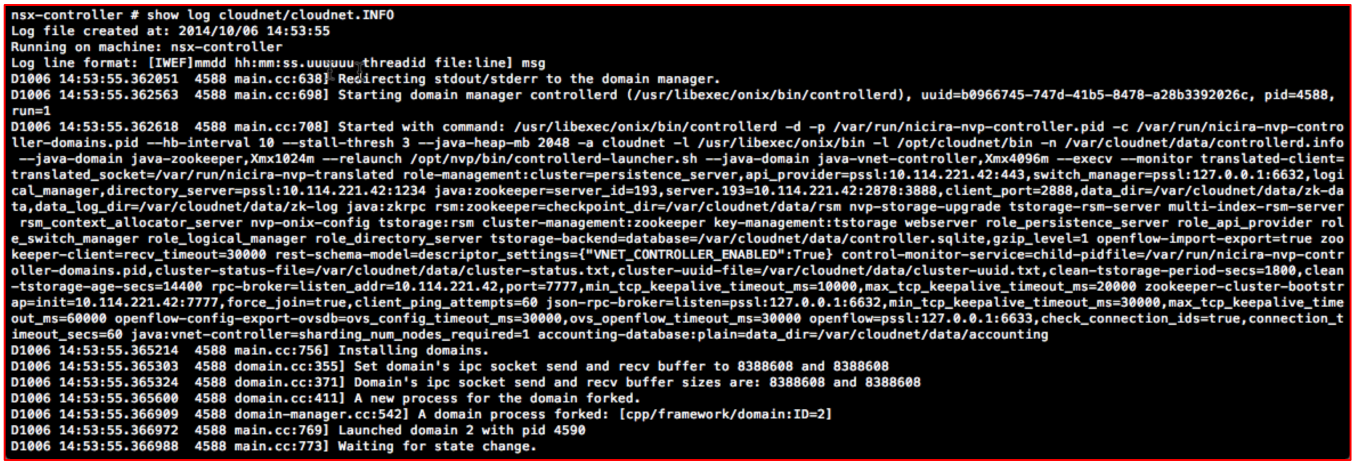


```
nsx-controller # show logs
S20python-carbon.err
S20python-carbon.log
S20python-graphite-web.err
S20python-graphite-web.log
alternatives.log
apt/history.log
apt/term.log
auth.log
boot
boot.log
btmp
cloudnet/cloudnet.INFO
cloudnet/cloudnet.nsx-controller.root.log.INFO.20141006-145355.4584
cloudnet/cloudnet.nsx-controller.root.log.INFO.20141006-145355.4588
cloudnet/cloudnet_cpp.log.INFO
cloudnet/cloudnet_cpp.log.nsx-controller.root.log.INFO.20141006-145355.4590
cloudnet/cloudnet_java-vnet-controller.20141006-145356.4650.log
cloudnet/cloudnet_java-zookeeper.20141006-145356.4622.log
cloudnet/cloudnet_python.log
cloudnet/csync2-client.log
cloudnet/log_backtraces.log
cloudnet/nicira-nvp-translated.log
cloudnet/nsx-controller.b0966745-747d-41b5-8478-a28b3392026c.20141006-145421.4590.monitorpairs.binlog
cloudnet/nsx-controller.b0966745-747d-41b5-8478-a28b3392026c.20141006-151952.4590.monitorpairs.binlog
cloudnet/nsx-controller.b0966745-747d-41b5-8478-a28b3392026c.20141006-154453.4590.monitorpairs.binlog
cloudnet/onix_heap_sample.4590.00000
cloudnet/onix_heap_sample.4590.00001
```

To view a log file,

show log <log file name>

Example:



```
nsx-controller # show log cloudnet/cloudnet.INFO
Log file created at: 2014/10/06 14:53:55
Running on machine: nsx-controller
Log line format: [IWEF]mmdd hh:mm:ss.uuuuuu threadid file:line] msg
D1006 14:53:55.362051  4588 main.cc:638] Redirecting stdout/stderr to the domain manager.
D1006 14:53:55.362563  4588 main.cc:698] Starting domain manager controllerd (/usr/libexec/onix/bin/controllerd), uuid=b0966745-747d-41b5-8478-a28b3392026c, pid=4588,
run=1
D1006 14:53:55.362618  4588 main.cc:708] Started with command: /usr/libexec/onix/bin/controllerd -d -p /var/run/nicira-nvp-controller.pid -c /var/run/nicira-nvp-contro
ller-domains.pid --hb-interval 10 --stall-thresh 3 --java-heap-mb 2048 -a cloudnet -l /usr/libexec/onix/bin -l /opt/cloudnet/bin -n /var/cloudnet/data/controllerd.info
 --java-domain java-zookeeper,Xmx1024m --relaunch /opt/nvp/bin/controllerd-launcher.sh --java-domain java-vnet-controller,Xmx4096m --execv --monitor translated-client=
translated_socket=/var/run/nicira-nvp-translated role-management:cluster=persistence_server,api_provider=pssl:10.114.221.42:443,switch_manager=pssl:127.0.0.1:6632,logi
cal_manager,directory_server=pssl:10.114.221.42:1234 java:zookeeper=server_id=193,server.193=10.114.221.42:2878:3888,client_port=2888,data_dir=/var/cloudnet/data/zk-da
ta,data_log_dir=/var/cloudnet/data/zk-log java:zkrpc rsm:zookeeper=checkpoint_dir=/var/cloudnet/data/rsm nvp-storage-upgrade tstorage-rsm-server multi-index-rsm-server
 rsm_context_allocator_server nvp-onix-config tstorage:rsm cluster-management:zookeeper key-management:tstorage webserver role_persistence_server role_api_provider rol
e_switch_manager role_logical_manager role_directory_server tstorage-backend=database=/var/cloudnet/data/controller.sqlite,gzip_level=1 openflow-import-export=true zoo
keeper-client=recv_timeout=30000 rest-schema-model=descriptor_settings={"VNET_CONTROLLER_ENABLED":True} control-monitor-service=child-pidfile=/var/run/nicira-nvp-contr
oller-domains.pid,cluster-status-file=/var/cloudnet/data/cluster-status.txt,cluster-uuid-file=/var/cloudnet/data/cluster-uuid.txt,clean-tstorage-period-secs=1800,clean
-tstorage-age-secs=14400 rpc-broker=listen_addr=10.114.221.42,port=7777,min_tcp_keepalive_timeout_ms=10000,max_tcp_keepalive_timeout_ms=20000 zookeeper-cluster-bootstr
ap=init=10.114.221.42:7777,force_join=true,client_ping_attempts=60 json-rpc-broker=listen=pssl:127.0.0.1:6632,min_tcp_keepalive_timeout_ms=30000,max_tcp_keepalive_time
out_ms=60000 openflow-config-export-ovsdb=ovs_config_timeout_ms=30000,ovs_openflow_timeout_ms=30000 openflow=pssl:127.0.0.1:6633,check_connection_ids=true,connection_t
imeout_secs=60 java:vnet-controller=sharding_num_nodes_required=1 accounting-database:plain=data_dir=/var/cloudnet/data/accounting
D1006 14:53:55.365214  4588 main.cc:756] Installing domains.
D1006 14:53:55.365303  4588 domain.cc:355] Set domain's ipc socket send and recv buffer to 8388608 and 8388608
D1006 14:53:55.365324  4588 domain.cc:371] Domain's ipc socket send and recv buffer sizes are: 8388608 and 8388608
D1006 14:53:55.365600  4588 domain.cc:411] A new process for the domain forked.
D1006 14:53:55.366909  4588 domain-manager.cc:542] A domain process forked: [cpp/framework/domain:ID=2]
D1006 14:53:55.366972  4588 main.cc:769] Launched domain 2 with pid 4590
D1006 14:53:55.366988  4588 main.cc:773] Waiting for state change.
```

The following log files on the Controller are of interest:

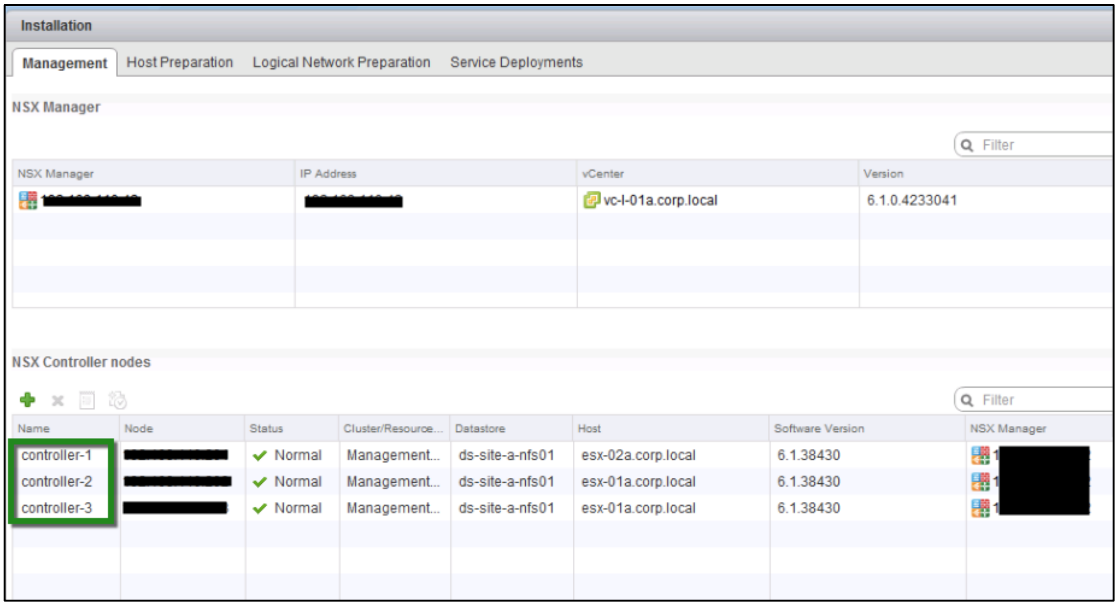
- General Controller logs: “cloudnet/cloudnet.INFO/WARNING/ERROR”
- REST API logs: “cloud/cloudnet_cpp.log”

NSX Controller logs to Remote Syslog server

System Events from controller can be sent to syslog server for analytics or archival. Syslog exporter can be enabled on the NSX Controller by using the published NSX Manager REST API.

```
Request:
POST https://<nsxmgr-ip>/api/2.0/vdn/controller/{controller-id}/syslog
Request Body:
<controllerSyslogServer>
<syslogServer>syslog-ip-address</syslogServer>
<port>514</port>
<protocol>UDP</protocol>
 <level>INFO</level>
</controllerSyslogServer>
```

The controller-id can be found from the NSX Manager Screen under NSX Home → Installation → Management



Installation

Management | Host Preparation | Logical Network Preparation | Service Deployments

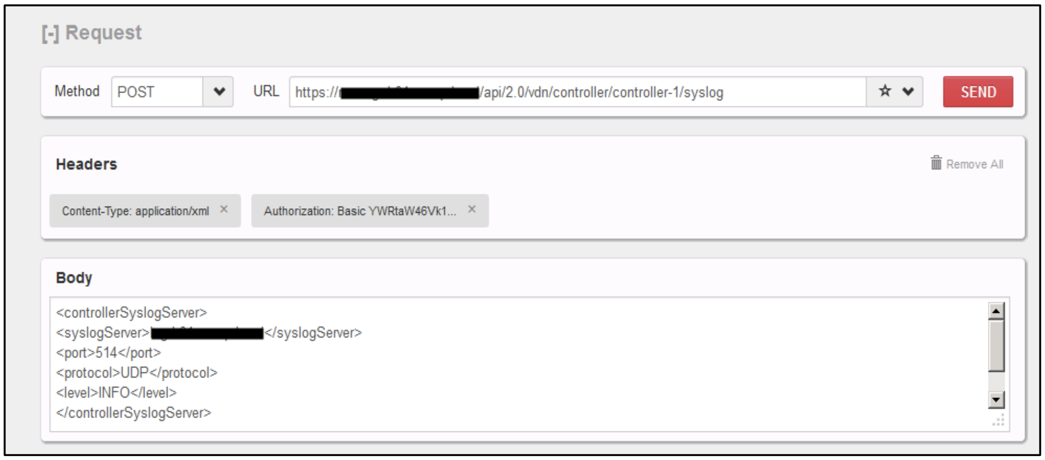
NSX Manager

NSX Manager	IP Address	vCenter	Version
[redacted]	[redacted]	vc-l-01a.corp.local	6.1.0.4233041

NSX Controller nodes

Name	Node	Status	Cluster/Resource...	Datastore	Host	Software Version	NSX Manager
controller-1	[redacted]	✓ Normal	Management...	ds-site-a-nfs01	esx-02a.corp.local	6.1.38430	[redacted]
controller-2	[redacted]	✓ Normal	Management...	ds-site-a-nfs01	esx-01a.corp.local	6.1.38430	[redacted]
controller-3	[redacted]	✓ Normal	Management...	ds-site-a-nfs01	esx-01a.corp.local	6.1.38430	[redacted]

Sample Request/Response: (In this example, we use the RESTClient plugin for Firefox)



Method POST

URL https://[redacted]/api/2.0/vdn/controller/controller-1/syslog

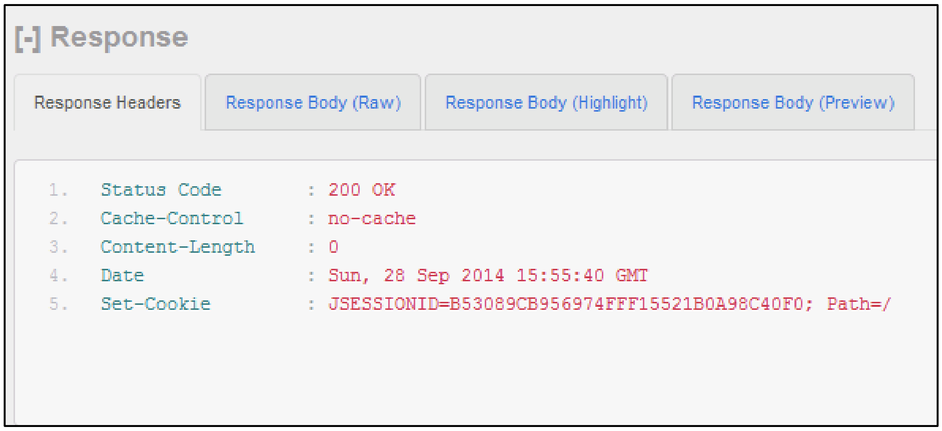
Headers

Content-Type: application/xml

Authorization: Basic YWRtaW46Vk1...

Body

```
<controllerSyslogServer>
<syslogServer>[redacted]</syslogServer>
<port>514</port>
<protocol>UDP</protocol>
<level>INFO</level>
</controllerSyslogServer>
```



```
Status Code        : 200 OK
Cache-Control      : no-cache
Content-Length     : 0
Date               : Sun, 28 Sep 2014 15:55:40 GMT
Set-Cookie         : JSESSIONID=B53089CB956974FFF15521B0A98C40F0; Path=/
```

Hypervisor Logs

Accessing Hypervisor Logs locally on the host: NSX Component logs such as Distributed Firewall, netcpa, Distributed Virtual Switch, can be viewed locally on each ESXi host.

List of log files related to NSX:

1. Distributed Firewall Packet logs can be found at **/var/log/dfwpktlogs.log**

2. Distributed Firewall UserWorld Agent logs: **/var/log/vsfwd.log**

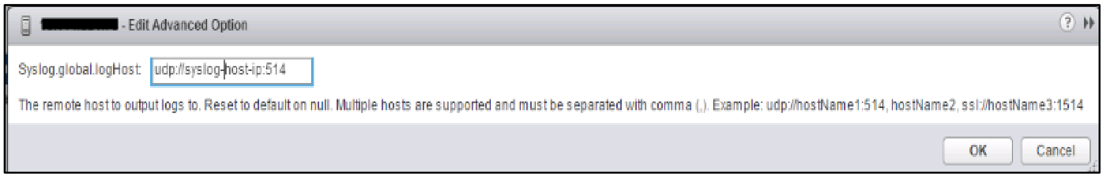
3. netcpa (User world agent) logs can be found at **/var/log/netcpa.log**. This log file will contain messages regarding controller to host communication details.

4. Logical Switch (VXLAN), Distributed Logical Routing (DLR) and VMware Internetworking Service Insertion Platform (**VSIP**) Kernel module logs are available at **/var/log/vmkernel.log**. The Logical Switch related logs will be tagged with **vxlan**, the Distributed Logical Router related logs will be tagged with **vdrb** and the VSIP related logs will be tagged with **vsip.**

5. DVS logs are also available at **/var/log/vmkernel.log**

Configuring Remote Syslog Server on ESXi hosts

Remote Syslog Server can be configured on a ESXi 5.x host using vSphere Web Client by editing the "`Syslog.global.logHost`" properties on the host. This property can be manually edited by browsing to Hosts -> [Select ESXi Host] -> Manage -> Settings -> Advanced System Settings on the vSphere Web Client.



[redacted] - Edit Advanced Option

Syslog.global.logHost: udp://syslog-host-ip:514

The remote host to output logs to. Reset to default on null. Multiple hosts are supported and must be separated with comma (,). Example: udp://hostName1:514, hostName2, ssl://hostName3:1514

To configure syslog server using ESXi Command line:

```
esxcli system syslog config set --loghost='udp://syslog-ip-addr:514'
```

or

```
esxcli system syslog config set --loghost='tcp://syslog-ip-addr:514'
```

Please refer to this KB article for other methods such as using Host Profiles for setting remote syslog server on multiple ESXi 5.x hosts.

http://kb.vmware.com/selfservice/microsites/search.do?language=en_US&cmd=displayKC&externalId=2003322

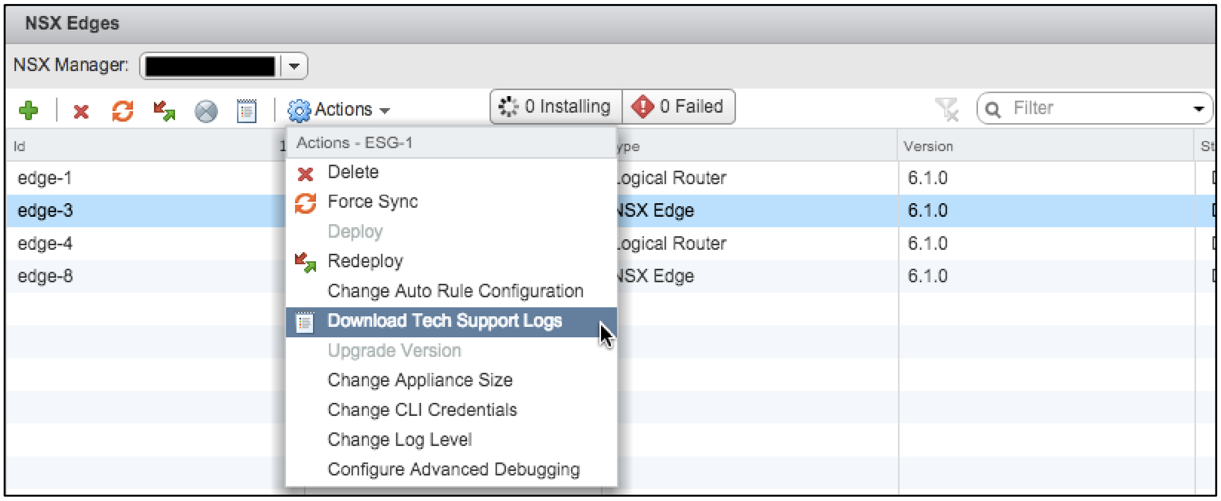
NSX Edge

NSX Edge is a Multi-function Virtual Services Appliance that provides various network services such as Layer 2 Bridging, Layer 3 Routing, Firewall, NAT, Load-Balancer, L2 VPN, IPSec VPN, Remote Access VPN and DHCP Service and DHCP Relay.

Downloading Tech Support Logs on NSX Edge

Tech Support Logs can be downloaded from the NSX Edge by browsing to NSX Home → NSX Edges → [Select NSX Edge] → Actions → Download Tech Support Logs

This will download a gz file on your local machine.



NSX Edges

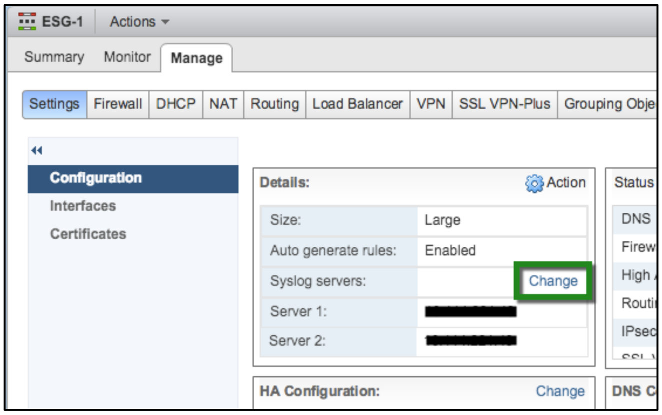
NSX Manager: [redacted]

Id	ype	Version
edge-1	ogical Router	6.1.0
edge-3	SX Edge	6.1.0
edge-4	ogical Router	6.1.0
edge-8	SX Edge	6.1.0

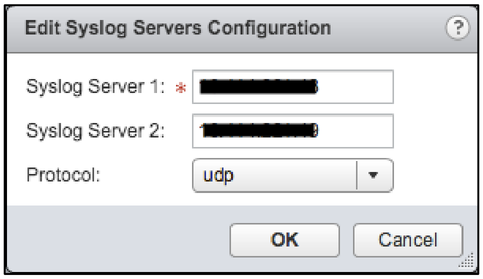
Enabling Remote Syslog on NSX Edge

Enabling NSX Edge to send syslog is a two-step process. The first step involves configuring the Syslog target. The second step is enabling logging and setting log level for each of the individual services mentioned above.

Step 1: To setup Syslog Target on the NSX Edge, browse to NSX Main → NSX Edge → Select/Double Click Edge → Manage → Settings → Configuration. Click **Change.** Add or Edit remote syslog target by entering FQDN or IP Address and select the protocol [UDP/TCP]. Click OK to save the settings. (Note that FQDN can be used only if DNS server is configured on the Edge.)



Size:	Large
Auto generate rules:	Enabled
Syslog servers:	Change
Server 1:	[redacted]
Server 2:	[redacted]



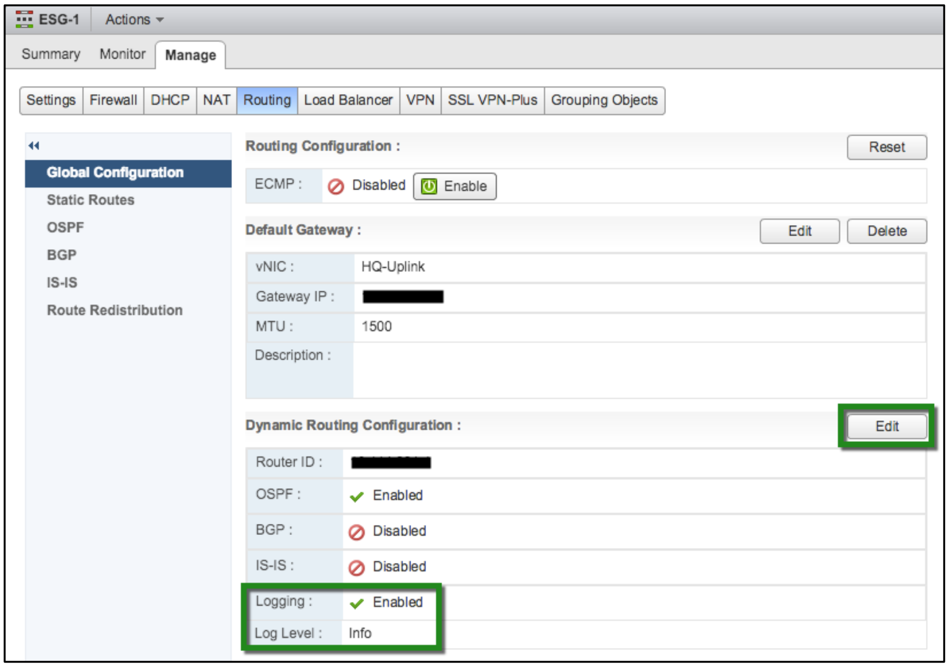
Syslog Server 1: * [redacted]

Syslog Server 2: [redacted]

Protocol: udp

Figure: Configuring Syslog Server

Step 2: To configure Log Level for a service, select the tab corresponding to that service and edit the Logging settings. In the example below, logging for Routing service is shown.



Routing Configuration :

ECMP : Disabled

Default Gateway :

vNIC : HQ-Uplink

Gateway IP : [redacted]

MTU : 1500

Description :

Dynamic Routing Configuration :

Router ID : [redacted]

OSPF : ✓ Enabled

BGP : Disabled

IS-IS : Disabled

Logging : ✓ Enabled

Log Level : Info

Figure: Setting Logging and Log Level for Individual Services

Backup and Restore

This section is intended for personnel who are interested in backup/restore procedures and failure scenarios for NSX for vSphere ("NSX-v").

Proper backup of NSX-v components is crucial to get the system back in working state in the event of a failure.

The NSX Manager backup contains all of the NSX configuration, including controllers, logical switching and routing entities, security, firewall rules, and everything else that you configure within the NSX Manager UI or API. The vCenter database and related elements like the virtual switches need to be backed up separately.

At a minimum, we recommend taking regular backups of NSX Manager and vCenter. Your backup frequency and schedule might vary based on your business needs and operational procedures. We recommend taking NSX backups frequently during times of frequent configuration changes.

At a minimum, we recommend taking regular backups of NSX Manager and vCenter to restore system state in the event of a catastrophic failure. Backup frequency and schedule may vary based on business need and operational procedures setup by operational teams. We recommend taking NSX backup frequently if there are frequent configuration changes happening.

NSX Manager backups can be taken on-demand or on an hourly, daily, or weekly basis.

In-order to recreate system state after a failure, we recommend taking backups in the following scenarios:

- Before an NSX or vCenter upgrade

- After an NSX or vCenter upgrade

- During/after Day Zero deployment and configuration of NSX components. (Creation of controllers, logical switches, distributed logical routers (DLR), Edge components, security and firewall policies)

- Infrastructure changes

- Any major Day-2 changes

To provide an entire system state at a given time to roll back to, we recommend synchronizing NSX component backups (such as NSX Manager) with your backup schedule for other interacting components, such as vCenter, cloud management systems, operational tools, and so on.

Individual Component Backups

NSX Manager Backup and Restore

You can back up NSX Manager data by performing an on-demand backup or a scheduled backup.

Backup/restore can be configured from the NSX Manager virtual appliance web interface or via API. Backups can be scheduled on an hourly, daily or weekly basis.

The backup file is saved out to a remote location that NSX-v manager can access via FTP or SFTP.

You can backup and restore your NSX Manager data, which can include system configuration, events, and audit log tables. Configuration tables are included in every backup.

Restore is only supported on the SAME NSX Manager version as the backup.

Back up the NSX Manager Configuration

1. Log in to the NSX Manager Virtual Appliance.
2. Under Appliance Management, click Backups & Restore.
3. To specify the backup location, click Change next to FTP Server Settings.
    a. Type the IP address or host name of the backup system.
    b. From the Transfer Protocol drop-down menu, select either SFTP or FTP, based on what the destination supports.
    c. Edit the default port if required.
    d. Type the user name and password required to login to the backup system.
    e. In the Backup Directory field, type the absolute path where backups will be stored.
    f. Type a text string in Filename Prefix.
    g. This text is prepended to each backup filename for easy recognition on the backup system. For example, if you type "ppdb", the resulting backup is named as ppdbHH_MM_SS_DayDDMonYYYY.
    h. Type the pass phrase to secure the backup. (You will need this passphrase to restore the backup).
    i. Click OK.
    j. *Note*: Please save your FTP server IP/hostname, credentials, directory details and pass-phrase. These will be needed to restore the backup.
4. For on-demand backup, click the Backup button as shown in the figure.
5. For scheduled backups, click Change next to Scheduling to pick a schedule for the backup. From the Backup Frequency drop-down menu, select Hourly, Daily, or Weekly. The Day of Week, Hour of Day, and Minute drop-down menus are disabled based on the selected frequency. For example, if you select Daily, the Day of Week drop-down menu is disabled as this field is not applicable to a daily frequency.
    - For a weekly backup, select the day of the week the data should be backed up.
    - For a weekly or daily backup, select the hour at which the backup should begin.
    - Select the minute at which the backup begins and click Schedule.
6. To exclude logs and certain items from being backed up, click Change next to Exclude. Select the items you want to exclude from the backup.

Restore the NSX Manager Configuration

It is recommended to restore a backup on a freshly deployed NSX Manager Appliance. Restore on an existing NSX Manager may work too but it is not officially supported/tested in-house. Internal testing is done with the assumption that the existing NSX Manager has failed hence a new NSX Manager appliance is deployed.

The freshly deployed NSX Manager appliance VM on which the restore is performed MUST be the same version as the NSX Manager appliance on which the backup was taken.

To restore an available backup, the Host IP Address, Username, Password, Backup Directory, Filename Prefix, and Passphrase fields in the Backup Location screen must have values that identify the location of the backup to be restored.

Best Practice: Take screenshots of the old NSX Manager Appliance settings screen or note them so that they can be used to specify IP information and backup location information on a freshly deployed NSX Manager appliance.

Procedure:

1. Deploy a fresh NSX Manager appliance VM. The version should be the same as the backed up NSX Manager appliance VM.
2. Login to the freshly deployed NSX Manager appliance VM.
3. Under Appliance Management, click Backups & Restore.
4. Go to FTP Server Settings, select Change and add the details of the backup location like IP/Hostname, Transfer Protocol, Port, Username, Password, Backup Directory, Filename Prefix and Passphrase.
5. In the Backups History section, select the check box for the backup to restore.
6. Click Restore.
7. Click OK to confirm.
8. You will need to re-login into the appliance.
9. Verify the NSX Management Service is running. You may need to wait a few minutes for the service to start running.
10. Verify the NSX Manager is connected to the vCenter & SSO services. Once the management service is running these should connect automatically.

NSX Edge Backup and Restore

All NSX Edge configurations (Logical Distributed Router control VMs and Edge Services Gateways) are backed up as part of an NSX Manager backup.

If the user has the NSX Manager configuration intact, the inaccessible or failed Edge appliance VMs can be redeployed anytime from the vSphere Web Client -> Networking and Security -> NSX Edges -> Actions -> Redeploy.

Taking individual NSX Edge backups is not supported.

If you want to retrieve the configuration of a standalone NSX Edge, you can use REST API calls to retrieve all the configuration. Details on how to use the REST API to manage NSX can be found in the *NSX vSphere API Guide*.

NSX Firewall Rule Backup and Restore

Export NSX Firewall Rules

You can export the Firewall rules configuration and save them to a central location. All firewall rules including Service Composer rules are exported. The saved configuration can be used as a backup or imported for use in an NSX Manager environment.

Procedure

1. Log in to the vSphere Web Client
2. Click Networking & Security and then click Firewall.
3. Go to the Configuration tab and to General.
4. Click the icon (the icon depicting a page with a small blue arrow at the lower right) and export configuration as shown in the figure.
5. Click the Download button to save the exported configuration file on the desktop as an XML file.
6. Select the directory where you want to save the file and click Save.

Note: All the sections are exported when you export a configuration (General, L2 and Partner Security services).

Import NSX Firewall Rules

1. Log in to the vSphere Web Client
2. Click Networking & Security and then click Firewall.
3. Go to the Saved Configurations tab.
4. Click the icon (the icon depicting a page with a small blue arrow at the upper left) to Import Configuration.
5. Click the Browse button to browse to the location where the exported firewall configuration file is stored and hit OK to import the file.
6. Hit Publish on the NSX Edge if the publish button appears.

Rules are imported based on rule names. During the import, the firewall ensures that each object referenced in the rule exists in your environment. If an object is not found, the rule is marked as invalid. If a rule references a dynamic security group, the dynamic security group is created in NSX Manager during the import.

When firewall rules are imported, the associated Service Composer rules are imported as well. If a security group does not exist in Service Composer, and an associated rule is imported as part of a firewall import, the system will flag that the associated security group doesn't exist.

When you load an imported firewall configuration, if your current configuration contains rules managed by Service Composer, these are overridden after the import.

If Service Composer rules in your configuration were overridden by the loaded configuration, click Actions > Synchronize Firewall Config in the Security Policies tab in Service Composer.

NSX Service Composer Backup and Restore

You can export a Service Composer configuration (along with the security groups to which the security policies are mapped) and save it to your desktop. The saved configuration can be used as a backup for situations where you may accidentally delete a policy configuration, or it can be exported for use in another NSX Manager environment.

Export Service Composer Configuration

Procedure

1. Log in to the vSphere Web Client.
2. Click Networking & Security and then click Service Composer.
3. Click the Security Policies tab.
4. Select the security policy that you want to export.
5. Click Actions and then click the Export Service Configuration icon.
6. Type a name and description for the configuration that you are exporting.
7. If desired, type a prefix to be added to the security policies and security groups that are being exported.
8. If you specify a prefix, it is added to the target security policy names thus ensuring that they have unique names.
9. Click Next.
10. In the Select security policies page, select the security policy that you want to export and click Next.
11. The Ready to complete page displays the security policies along with associated objects (security groups on which these have been applied, as well as Endpoint services, firewall rules, and network introspection services) to be exported.
12. Click Finish.
13. Select the directory on your computer where you want to download the exported blueprint and click Save.

The configuration file is saved at the specified location.

Import a Service Composer Configuration

You can import a saved Service Composer configuration (along with the security groups to which the security policies are mapped) either as a backup or to restore configuration on a different NSX Manager.

Procedure

1. Log into the vSphere Web Client
2. Click Networking and Security and then click Service Composer
3. Click the Security Policies tab.
4. Click Actions and then click the Import Service Configuration icon.
5. Select the configuration file that you want to import.
6. If desired, type a suffix to be added to the security policies and security groups that are being imported. If you specify a suffix, it is added to the security policy names being imported thus ensure that they have unique names.
7. Click Next.
8. Service composer verifies that all services referred to in the configuration are available in the destination environment. If not, the Manage Missing Services page is displayed, where you can map missing services to available target services.

9. The Ready to complete page displays the security policies along with associated objects (security groups on which these have been applied, as well as Endpoint services, firewall rules, and network introspection services) to be imported.
10. Click Finish.

The imported security policies are added to the top of the security policy table (above the existing policies) in the targeted NSX Manager. The original order of the imported policies is preserved.

Virtual Distributed Switch Backup and Restore

Export a Distributed Switch Configuration

You can export vSphere distributed switch and distributed port group configurations to a file. The file preserves valid network configurations, enabling distribution of these configurations to other deployments. This functionality is available only with the vSphere Web Client 5.1 or later. VDS settings and port-group settings are imported as part of this import. As a best practice, you should export the VDS configuration before preparing the cluster for VXLAN.

Procedure

To export the vSphere Distributed Switch configuration using the vSphere Web Client, follow these steps:

1. Browse to a distributed switch in the vSphere Web client navigator
2. Right-click the distributed switch and click All vCenter Actions > Export Configuration.
3. Select the Export the distributed switch configuration or Export the distributed switch configuration and all port groups option.
4. (Optional) Enter notes about this configuration in the Description field.
5. Click OK.
6. Click Yes to save the configuration file to your local system.

You now have a configuration file that contains all settings for the selected distributed switch and distributed port group. You can use this file to create multiple copies of this configuration on an existing deployment or overwrite the settings of existing distributed switches and port groups to conform to the selected settings.

Import vSphere Distributed Switch

Procedure

1. Browse to a distributed switch whose configuration you need to restore in the vSphere Web client navigator. It could be an existing switch with some configuration or a brand new distributed switch that was created just to import the configuration.
2. Right-click the distributed switch and click All vCenter Actions > Restore Configuration.
3. Browse for the configuration backup file to use.
4. Select the Restore distributed switch and all port groups or Restore distributed switch only option and click Next.
5. Review the summary information for the restore.
6. Click Finish.

vCenter Backup and Restore

Please refer to the vCenter documentation for your vCenter version for vCenter backup and restore procedures and best practices. The vCenter documentation covers this:

- Back up vCenter Server 5.1
- Back up vCenter Server 5.5
- Restore vCenter Server 5.1
- Back up and restore VMware vCenter Single Sign-On 5.x
- Back up and restore the vCenter Server 5.x database
- Back up and restore the vCenter Server 6.x database
- White paper: http://www.vmware.com/files/pdf/techpaper/vmware-vcenter-server-availability-guide.pdf

Failure and Recovery Scenarios

A typical NSX design is architected with high availability, resiliency and recovery. In a highly redundant and available system, chances of failure are rare. There are the various protection mechanisms in place for various components of the NSX system. The following are the recommended mechanisms to protect each NSX component.

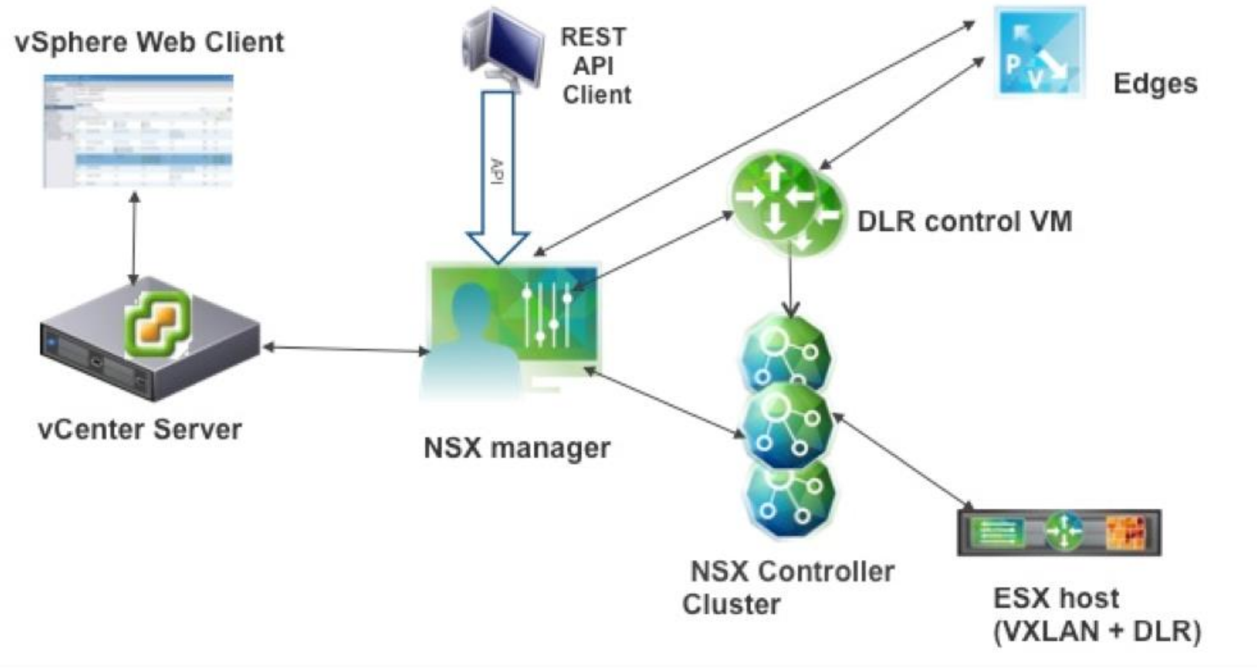
- NSX Manager: Use vSphere HA.
- NSX Controllers: Maintain a controller cluster size of 3.
- NSX DLR Control VM: Use Edge HA.
- NSX Edge Services Gateway: Use Edge HA.
- VMware vCenter: Various DB replication mechanisms.

More information on highly resilient and available NSX design can be found in our Reference Architecture guide at: http://www.vmware.com/files/pdf/products/nsx/vmw-nsx-network-virtualization-design-guide.pdf

In the rare event of a failure, recovery of components and full state sync between different components is crucial. Before going into failure and out-of-sync scenarios, it is important to understand the NSX architecture and how different components interact with each other.

NSX Architecture and Component Interaction

NSX Manager virtual appliance is the management plane entry-point for configuring and interacting with all the NSX components. NSX Manager has tight integration with the vCenter that is used to manage all the compute components.



VXLAN + DLR Communication

As shown in the figure, NSX Manager is serviceable by either the NSX Manager UI (configurable through Networking and security plugin in the vSphere Web Client) or the REST API. The following diagram describes the communication path.

Communication Path: VXLAN + DLR + Edges

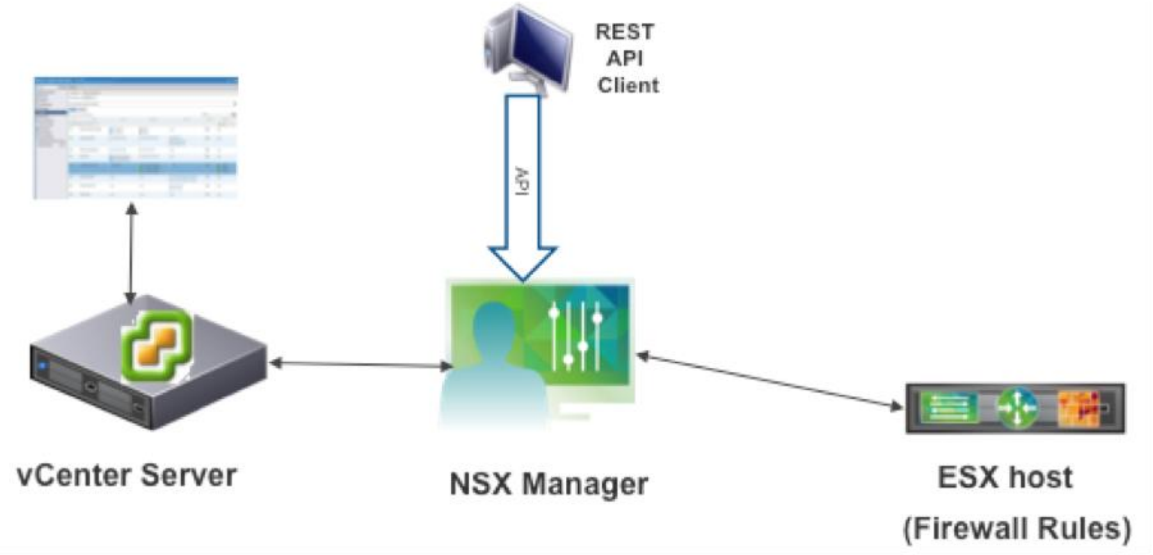
For logical network configuration (L2), the NSX Manager sends logical network (VNI) information to the controller cluster, and the controller cluster propagates this information to the ESXi hosts.

For routing (L3), the NSX Manager configures the DLR control VM and the Edges. The DLR control VM and the Edges form routing adjacencies and exchange route information. The learnt routes are distributed to the controller cluster, which then distributes the route updates to the ESXi hosts.

If NSX Manager is restored from backup, the information from NSX Manager needs to be synced to the controllers, the DLR control VMs, the NSX Edges and to the hosts.

Propagation of Firewall Rules

Firewall rules can be configured by using the Firewall Menu or the Service Composer Menu in the Networking and Security plugin in the vSphere Web Client (or via Rest API). The following diagram describes the communication path for the propagation of firewall rules. Firewall rules are directly pushed from NSX Manager to the ESXi hosts that are part of the NSX cluster.



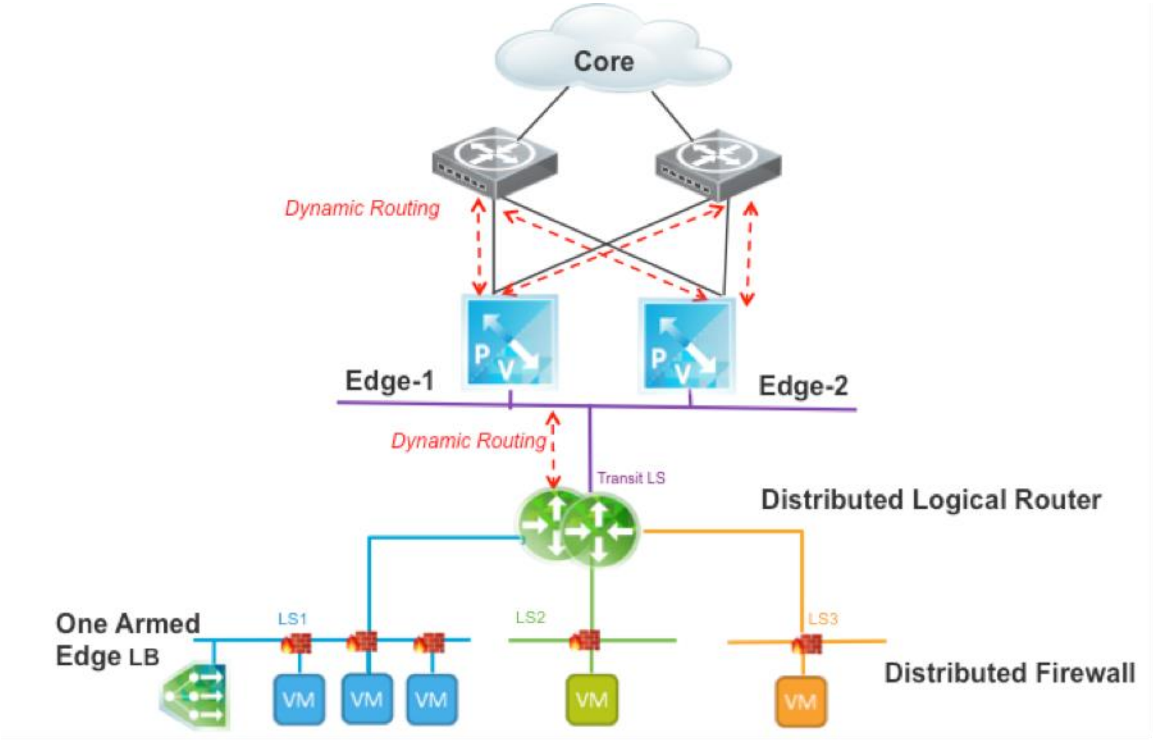
Communication Path: Propagation of firewall rules.

When the NSX Manager is restored from backup, the firewall rule information from the NSX Manager is automatically synced to the corresponding ESXi hosts.

Example Failure Scenarios

This section covers various failure scenarios and how to recover from each of them. This section assumes total component failure (of both master and HA partner). We will attempt to address some select failure scenarios here. Each scenario will assume the following components are configured and in working condition at initial state.

Example topology for failure scenarios



As a base topology, there is a fully functional NSX environment with

- 3 controllers
- 4 logical switches: LS1, LS2, LS3, Transit LS
- One DLR Control VM (in Edge HA)
- 2 NSX Edges in ECMP
- 1 One Armed Edge LB
- Routing protocol is running between DLR and Edge VMs.
- Routing protocol is running between Edge VMs and TOR switches.
- ECMP is enabled on both the DLR and on the Edges.
- Security groups, security policies and firewall rules in place.
- There are VMs running on LS1, LS2 and LS3

The examples assume the following backups have been taken and stored at a centralized location:

- vCenter (database backup / VM snapshot)
- NSX Manager

Scenario 1: Single NSX Controller VM Failure

This scenario describes how to recover from a single controller VM failure. In case of a single NSX Controller VM Failure, we still have two controllers which are working so cluster majority is maintained and the control plane will continue to function.

Recovery Procedure

1. Log in to vSphere Web Client.

2. Go to Networking and Security Tab and click Installation and ensure the Management tab is selected.
3. In the NSX Controller nodes section, find the find failed/disconnected controller. Take a screenshot/print-screen of the screen or write down the configuration information for each of the controllers so you can refer to them later.
4. In the NSX Controller nodes section, find the failed controller and delete it by selecting it and clicking the Delete Node (x) icon.
5. Deploy a new NSX Controller Node, click the Add Node (+) icon.
6. In the Add Controller dialog box, select the datacenter on which you are adding the node and configure the controller settings.
    a. Select the appropriate cluster.
    b. Select a Host in the cluster and storage.
    c. Select the distributed port-group.
    d. Select the IP pool from which IP addresses are to be assigned to the node.
    e. Click OK, Wait for installation to complete and ensure all nodes have a status of Normal.

Scenario 2: NSX Controller Cluster Failure

This scenario assumes total failure of NSX Controller Cluster (all 3 controller VMs are inaccessible). In the event of an NSX Controller Cluster failure, the data plane will continue running headless. To recover the controllers, we will redeploy the controllers, re-sync VXLAN and redeploy DLR control VMs.

Recovery Procedure

1. Log in to vSphere Web Client
2. Go to Networking and Security Tab and click Installation and ensure the Management tab is selected.
3. In the NSX Controller nodes section, find the find failed/disconnected controllers. Take a screenshot of the screen or write down the configuration information for each of the controllers so you can refer to them later.
4. In the NSX Controller nodes section, find the failed controllers and delete them by selecting them, and clicking the Delete Node (x) icon.
5. Redeploy all the 3 controllers again by following the procedure to Add Node (x) for each controller as follows:
6. Deploy a new NSX Controller Node, click the Add Node (+) icon.
7. In the Add Controller dialog box, select the datacenter on which you are adding the node and configure the controller settings based on the configuration information of the deleted controller.
    a. Select the appropriate cluster.
    b. Select a Host in the cluster and storage.
    c. Select the distributed port-group.
    d. Select the IP pool from which IP addresses are to be assigned to the node.
    e. Click OK, Wait for installation to complete and ensure all nodes have a status of Normal.

If all three NSX Controllers are lost or deleted, please follow the sync procedure articulated in the following KB article to sync NSX Manager to the new NSX Controllers and the NSX Edges:

http://kb.vmware.com/selfservice/microsites/search.do?language=en_US&cmd=displayKC&externalId=2112838

Scenario 3: NSX Edge Failure

This scenario assumes loss of Edge VMs and their HA counterparts. When the Edge VMs and their HA partners are both lost, a data plane outage is expected.

This procedure can be used to recover all Edge VMs or only selected VMs. The same procedure can also be used to recover DLR control VMs. The assumption is that NSX Manager and the NSX Controllers are intact and in working condition in this scenario. Hence, this scenario assumes that the configuration for all the Edges is present and accessible in the NSX Manager database (that is, you can view the Edge configuration from the NSX Manager Networking and Security plugin of the vCenter web client or by querying the configuration from the REST API).

Recovery Procedure

1. Log in to the vSphere Web Client.
2. Click Networking & Security and then click NSX Edges.
3. Locate the NSX Edge Type Edge Device.
4. Right click the NSX Edge and select Redeploy. (In the event of a datastore or host failure you may want to change the Edge settings before redeploying the same).
5. Select vCenter, Hosts, NSX Cluster, locate the Edge Services Gateway Virtual Machines, check to see if they are powered on.

The redeploy procedure will not delete the inaccessible Edge VMs. As administrator, you must manually delete those VMs from the inventory once those VMs become accessible.

Scenario 4: NSX Prepared Host Failure

This scenario covers the case where an NSX prepared host fails and loses all its networking information. In this scenario:

- The host is part of NSX Compute Cluster and part of a transport zone.

- The host comes back online.

Recovery procedure

This recovery procedure assumes network settings are intact. In this case, no user intervention is needed. You can check for errors by going to vSphere Web Client -> Network & Security -> Installation-> Host Preparation and checking if there are errors on any of the hosts/clusters. If there are errors, they can generally be resolved by clicking the Force Sync Configuration on the VXLAN.

Scenario 5: vCenter Failure

In this scenario, the vCenter Server Appliance has experienced an issue due to which it is reverting to an older copy of vCenter database. This scenario assumes that all NSX components in the system are working and only the vCenter is down. It also assumes that we have a backed up vCenter Database snapshot from which we can restore. This scenario assumes that a few new Logical Switches and/or VMs have been added since the last backup. As a result, at the time of the failure, NSX Manager is aware a new logical switch and VM(s) attached to it, but the VC backup does not contain records of the new logical switch and VM(s). The following steps should be taken to normalize the environment after this failure.

Here we outline the base scenario, starting at time "t_0":

At time t_0, we have a working NSX and vCenter environment.

At time t_1, the backup of vCenter database is taken. A backup of NSX Manager is also taken.

Next, we introduce changes: At time t_2, a logical switch "LS-4" is added and VMs are added to this logical switch.

At time t_3, a failure occurs, and the vCenter database needs to be restored due to this failure. NSX continues to run because the hosts housing all NSX components and VMs remain operational.

At time t_4, we initiate recovery/restore of vCenter. Follow the recovery procedure below for instructions.

Recovery Procedure

1. Restore the vCenter Database. Follow vCenter Server Appliance database restore instructions to restore the database:
    a. If running the VMware vCenter Appliance (VCSA) restore the VC VM and point it to the restored DB. Follow the vCenter backup/restore instructions.
    b. If running vCenter on MS Windows, verify the database connection is successfully connecting to the restored database. Follow the vCenter backup/restore instructions.
    c. Take care of SSO considerations according to the vCenter backup/restore instructions.
2. Use NSX Manager to check that all needed port groups are present:
    a. Wait until the vCenter is up and running, and then log in to the NSX Manager appliance management GUI, and go to Manage -> NSX Management Service and check if the vCenter Server connection is intact. Refresh the connections if needed.
    b. Once vCenter is up and running, check if the Networking and Security is visible as a plug-in in the vSphere Web Client.
    c. Browse through the Networking and Security plugins and check if the Logical switches are visible.
    d. At this point vCenter will have the Virtual Distributed switches missing the port group created for the extra logical switch LS-4 in NSX. Any VM on the logical switch will show that they are connected to port group that has no available data.
    e. The NSX Manager GUI in vCenter will show the extra logical switch in “Alert” mode next to it.
3. Next, follow the repair procedure outline below. In order to repair the logical switch LS-4 and re-sync/republish it as a VXLAN based port group to all the Virtual Distributed switches, you will run the "action=repair" REST API command:

```
POST https://<nsxmgr-ip>/api/2.0/vdn/scopes/vdnscope-id?action=repair
```

Before you can run "action=repair", you must get the VDN scope-id using the "scopes" REST API command:

```
GET https://<nsxmgr-ip>/api/2.0/vdn/scopes
```

The response body will be similar to:

```
<?xml version="1.0" encoding="UTF-8"?>

<vdnScopes>

 <vdnScope>
```

```
<objectId>vdnscope-1</objectId>
<objectTypeName>VdnScope</objectTypeName>
<vsmUuid>423F5BF5-E791-682A-AD6F-8E8BB4C2649E</vsmUuid>
<revision>0</revision>
<type>
   <typeName>VdnScope</typeName>
</type>
<name>TZ</name>
<description />
<clientHandle />
<extendedAttributes />
<id>vdnscope-1</id>
<clusters>
   <cluster>
      <cluster>
         <objectId>domain-c261</objectId>
         <objectTypeName>ClusterComputeResource</objectTypeName>
         <vsmUuid>423F5BF5-E791-682A-AD6F-8E8BB4C2649E</vsmUuid>
         <revision>10</revision>
         <type>
            <typeName>ClusterComputeResource</typeName>
         </type>
         <name>Cluster_New_edge</name>
         <scope>
            <id>datacenter-2</id>
            <objectTypeName>Datacenter</objectTypeName>
            <name>DC-REF</name>
         </scope>
         <clientHandle />
         <extendedAttributes />
      </cluster>
   </cluster>
      <cluster>
         <cluster>
         <objectId>domain-c22</objectId>
```

```
             <objectTypeName>ClusterComputeResource</objectTypeName>
             <vsmUuid>423F5BF5-E791-682A-AD6F-8E8BB4C2649E</vsmUuid>
             <revision>14</revision>
             <type>
                <typeName>ClusterComputeResource</typeName>
             </type>
             <name>Cluster_Compute_1</name>
             <scope>
                <id>datacenter-2</id>
                <objectTypeName>Datacenter</objectTypeName>
                <name>DC-REF</name>
             </scope>
             <clientHandle />
             <extendedAttributes />
          </cluster>
       </cluster>
    </clusters>
    <virtualWireCount>6</virtualWireCount>
    <controlPlaneMode>UNICAST_MODE</controlPlaneMode>
 </vdnScope>
</vdnScopes>
```

From the GET response-body above, you can determine that object-id for vdnScope (transport-zone) is "vdnscope-1".

4. Run the "action=repair" REST API call to repair vdnscope-1.

```
POST https://<nsxmgr-ip>/api/2.0/vdn/scopes/vdnscope-1?action=repair
```

The repair should be complete. Check it as follows:

Go to vSphere Web Client -> Networking and Security -> Logical switches and checking if the alert next to the logical switch LS-4 has disappeared. If it still appears, refresh the screen.

Go to vSphere Web Client -> Networking tab and check all the Virtual Distributed switches that are part of the transport zone associated with this logical switch.

All the VDS's in the transport zone should now have a VXLAN port group that maps with the logical switch LS-4 and is associated with its VNI. Typically the nomenclature of this port group is:

```
vxw-dvs-xxx-xx-sid-<vni-id>-<logical-switch-name>
```

Check that the VMs are correctly attached to this logical switch.

Scenario 6: NSX Manager Failure (DFW rules were added after backup)

This scenario is an NSX Manager failure scenario. In this example, we assume some DFW rules were changed after NSX Manager backup was taken, and we assume that no other changes were made. Here we outline the base scenario, starting at time "t_0":

At time t_0. NSX Manager was backed up.

At time t_1, we introduce changes by adding DFW rules to the firewall rule table.

At time t_2, NSX Manager Fails

Recovery Procedure

Restore NSX Manager by following the restore NSX Manager instructions from in the section, "NSX Manager Backup and Restore."

After the restore is complete, check the current state of firewall rules and replace any that are missing. Since firewall rules are directly pushed from NSX Manager to the hosts, the DFW rules from the restored NSX Manager snapshot will be automatically pushed to the ESXi hosts. This overwrites existing rules on the hosts, meaning that any rules that were added to the NSX Manager after the backup was taken are now lost. You must reconfigure those rules on the restored NSX Manager.

To verify that the rules are correctly pushed to the ESXi hosts, pick an ESXi host that hosts a VM to which the policy is applied. Run the following CLI commands in the ESX shell to find the VM UUID and the filter name on which the firewall rules are applied. For example:

```
# vsipioctl getfilters

Filter Name              : nic-12490976-eth0-vmware-sfw.2

VM UUID                  : 50 3b 68 1a 7f a0 bd 76-fc 77 e0 a2 59 e2 5a 61

VNIC Index               : 0

Service Profile          : --NOT SET--
```

- Run the following command to look at the rules associated with this filter. For example:

```
vsipioctl getrules -f nic-12490706-eth0-vmware-sfw.2

ruleset domain-c22 {

  # Filter rules

  rule 1006 at 1 inout protocol icmp icmptype 8 from addrset ip-securitygroup-12 to addrset
ip-securitygroup-12 reject with log;

  rule 1005 at 2 inout protocol any from addrset ip-securitygroup-11 to addrset ip-
securitygroup-11 accept with log;

}
```

If needed, run Force Sync Firewall at the host/cluster level from the vSphere Web Client -> Networking and Security plugin -> Host preparation tab by clicking on the Firewall-> Enabled and selecting Force Sync as shown in the figure.

Troubleshooting Backup and Restore Operations

The following commands will help the user verify system-state after any restore event.

Controller Troubleshooting for Backup and Restore

Verify if the controller-cluster has majority: `show control-cluster status`

Verify the controller nodes: `show control-cluster startup-nodes`

Verify if logical-switches are programmed in controller: `show control-cluster logical-switches vni <vni-id>`

Verify if the logical-switch host connection table: `show control-cluster logical-switches connection-table vni <vni-id>`

Verify if logical-router information is programmed in controller: `show control-cluster logical-router instance all`

ESXi Host Level Troubleshooting for Backup and Restore

Find VDS name associated with VTEP: `esxcli network vswitch dvs vmware vxlan list`

Verify logical network information and controller-plane connection per logical-switch: `esxcli network vswitch dvs vmware vxlan network list --vds-name <VDS_Name>`

Verify controller connection from host: `/etc/init.d/netcpad <status/start/stop/restart>`

Verify the firewall process running on the host: `/etc/init.d/vShield-stateful-firewall <status/start/stop/restart>`

Display firewall rules associated with a VM vnic: `vsipioctl getrules -f <filter-name>`

User Management

NSX integrates with VMware SSO services allowing for direct integration with Microsoft AD & LDAP user stores. With SSO, NSX supports authentication using authenticated Security Assertion Markup Language (SAML) tokens from a trusted source via REST API calls. NSX Manager can also acquire authentication SAML tokens for use with other VMware solutions.

Managing User Rights

NSX-v supports system defined role based access control (RBAC) as a means of regulating access to resources to users based on a user’s group membership. Within NSX-v there are four defined roles however it’s important to note a few requirements:

A user may only have one role.

You cannot add a role to a user or remove an assigned role from a user. You can, however, change the assigned role for a user.

NSX Manager User Roles

Role	**Permission / Entitlement**
Enterprise Administrator	NSX operations and security.
NSX Administrator	NSX operations only: for example, install virtual appliances, configure port groups
Security Administrator	NSX security only: for example, define data security policies, create port groups, and create reports for NSX modules.
Auditor	Read only

Managing User Accounts

Important Note! Each NSX virtual appliance has a built-in CLI user account ("nobody") for system use. Do not delete or modify this account. If this account is deleted or modified, the virtual machine for that appliance will not work.

You will typically set up vCenter SSO to authenticate against an Active Directory (AD) server with user groups corresponding to the roles listed here:

- Enterprise Administrator
- NSX Administrator
- Security Administrator
- Auditor

Once you have defined AD user groups that correspond to the above roles and connected to vCenter SSO, place each administrative user account in the appropriate group and he/she will inherit the appropriate NSX permissions. All of such groups have a global scope in NSX. If you require fine-grained access control beyond these groups, please refer to the user management section of the VMware NSX Documentation.

NSX Manager User Accounts

Important Note! You can manage the NSX Manager Appliance *admin* user only through the CLI commands.

Managing User Permissions

- Users are not created in NSX Manager rather NSX roles are mapped to existing vCenter & Active Directory users and groups
- If you delete a vCenter "user" account, only the role assignment for NSX Manager is deleted. The user account in vCenter or Active Directory is not deleted.

Instructions for the following actions can be found in the NSX for vSphere Administration Guide

Action	**Description**
Assign a role to a SSO user	Adds a user or group to a predefined role inside NSX
Delete a User account	Removes a user or group from all NSX roles

Change the NSX Manager *admin* User Account Password

Procedure

1. Log in to the vSphere Client and select an NSX virtual appliance from the inventory

2. Click the Console tab to open a CLI session.

3. Log in to the CLI and switch to Privileged mode.

4. At the manager prompt, type “enable” and provide the current password:

```
manager> enable

password:

manager#
```

5. Switch to Configuration mode.

```
manager# configure terminal
```

6. Use the “cli password” command to change the NSX Manager *admin* account password.

```
manager(config)# cli password PASSWORD
```

where PASSWORD is replaced with the new password you want to use.

7. Save the configuration.

```
manager(config)# write memory

Building Configuration...

Configuration saved.

[OK]
```

Change the NSX Manager *admin* User Account Privileged Mode Password

1. Log in to the vSphere Client and select an NSX virtual appliance from the inventory.

2. Click the Console tab to open a CLI session.

3. Log in to the CLI and switch to Privileged mode.

4. At the manager prompt, type “enable” and provide the current password:

```
manager> enable

password:

manager#
```

5. Switch to Configuration mode.

```
manager# configure terminal
```

6. Use the “enable password” command to change the *Privileged mode* password.

```
manager(config)# enable password PASSWORD
```

7. Save the configuration.

```
manager(config)# write memory

Building Configuration…

Configuration saved.

[OK]
```

8. Run the exit command twice to log out of the CLI.

```
manager(config)# exit

manager# exit
```

9. Log in to the CLI and switch to Privileged mode by using the new password.

```
manager> enable

password:

manager#
```

NSX Manager Password recovery

If you lose the NSX Manager *admin* password, contact VMware support for assistance resetting it.

NSX Controller User Accounts

NSX Controller Password Recovery

The NSX Controller *admin* account password is defined when the first NSX Controller appliance is created. Subsequent NSX Controllers inherit the configuration from the first. If you lose the NSX Controller *admin* password, contact VMware support for assistance resetting it.

Edge Appliance User Accounts

Change or Reset NSX Edge Appliance Password

NSX Edge appliances have a default user, *admin*, with a password that is defined at creation time. In a vCAC-initiated creation of an NSX Edge, the password may not have been changed and may still be set to “default”. In this event the Edge password should be manually changed to a secure value, and you should change your NSX Edge creation workflow in vCAC to ensure the password is changed at creation time.

Reset/Change Password from UI

Procedure to reset or change the admin password on an NSX Edge:

1. In Networking & Security click on NSX Edges.

2. Highlight the NSX Edge whose *admin* password you want to change.

3. In the Actions drop down menu above click on “Change CLI credentials.”

Reset/Change Password from API

NSX Edge passwords may also be reset via the API. A bulk reset of NSX Edge passwords is possible by doing the following

1. Query all NSX Edges with the following GET:

```
GET https://<nsxmgr-ip>/api/4.0/edges/
```

2. Loop thru the Modify User method for all NSX Edge users whose passwords you wish to change

```
PUT https://<vsm-ip>/api/4.0/edges/<NSX-Edge-Id>/sslvpn/config/auth/localserver/users/
```

Appendix A: Preparing to use the NSX REST API

The VMware NSX Network Virtualization Platform provides a programmatic API to automate operations and monitor events of logical networks in a virtualized hosting environment. Clients interact with the API using RESTful web service calls. You can invoke REST API calls using the web browser "RESTClient" or similar plug-in for browsers like Firefox and Chrome, or you can use the curl utility at the command line.

Configuring RESTClient for Firefox and Chrome web browsers

Note: To make XML responses more legible, you can copy and paste them into an XML friendly editor such as xmlcopyeditor or pspad.

To use the REST API in Firefox

1. Locate the RESTClient Mozilla add-on, and add it to Firefox.
2. Click Tools > REST Client to start the add-on.
3. Click Login and enter the NSX login credentials, which then appear encoded in the Request Header.
4. Select a method such as GET, POST, or PUT, and type the URL of a REST API. You might be asked to accept or ignore the lack of SSL certificate. Click Send.

Response Header, Response Body, and Rendered HTML appear in the bottom window.

To use the REST API in Chrome

Search the Web to find a REST Client, and add it to Chrome. (for example, DHC, Advanced REST client)

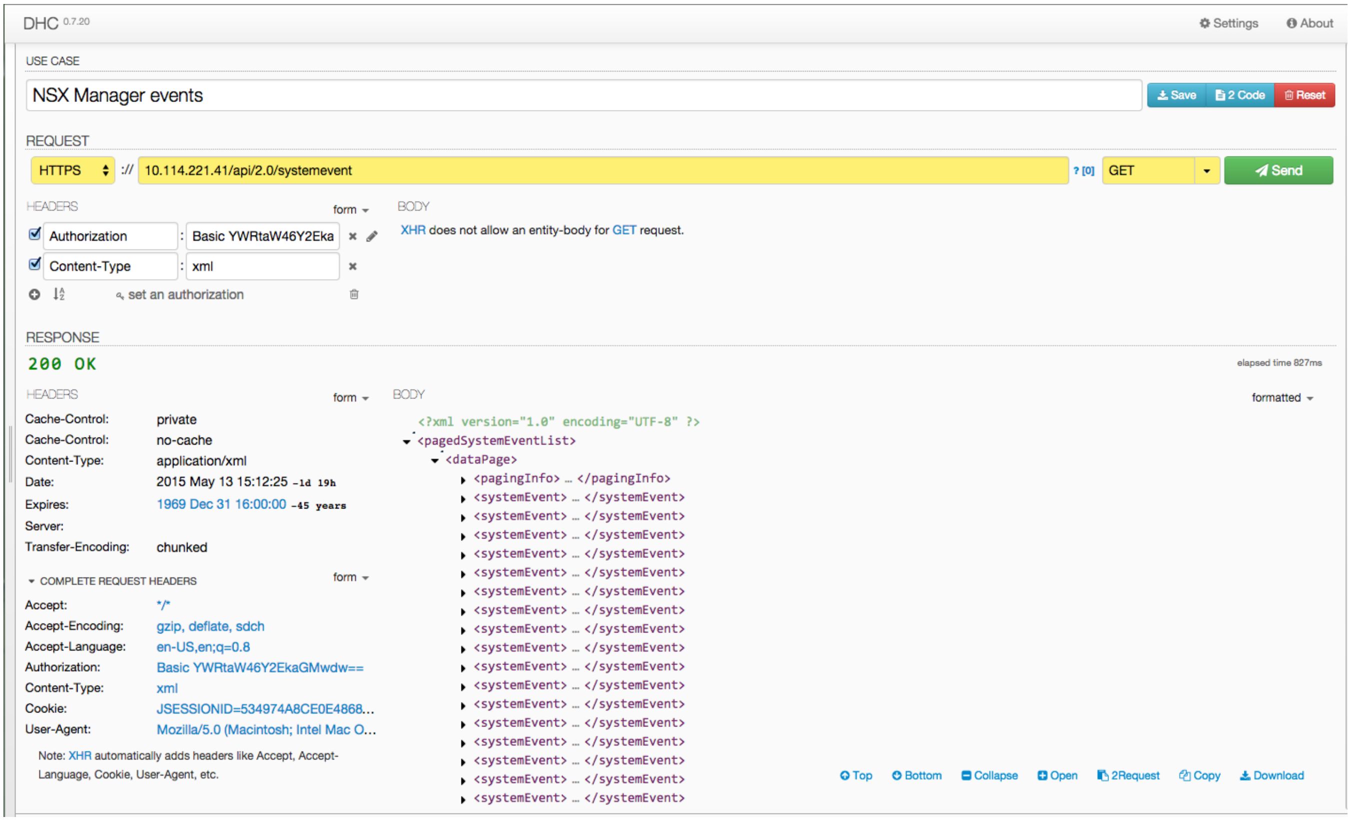
1. Click its globe-like icon to start it in a tab.
2. The Simple REST Client provides no certificate-checking interface, so use another Chrome tab to accept or ignore the lack of SSL certificate.
3. Type the URL of a REST API, and select a method such as GET, POST, or PUT.
4. In the Headers field, type the basic authorization line, as in the important note above. Click Send.

Status, Headers, and Data appear in the Response window.

In the examples below, 10.114.221.41 is the IP Address of NSX Manager.

Given below is a screenshot where Chrome web browser is used with DHC REST/HTTP API client to make the following API call to NSX Manager.

```
API call: GET https://10.114.221.41/api/2.0/systemevent
```



USE CASE

NSX Manager events

REQUEST

HTTPS :// 10.114.221.41/api/2.0/systemevent GET

HEADERS

☑ Authorization : Basic YWRtaW46Y2Eka

☑ Content-Type : xml

BODY

XHR does not allow an entity-body for GET request.

RESPONSE

200 OK

HEADERS

Cache-Control: private
Cache-Control: no-cache
Content-Type: application/xml
Date: 2015 May 13 15:12:25 -1d 19h
Expires: 1969 Dec 31 16:00:00 -45 years
Server:
Transfer-Encoding: chunked

COMPLETE REQUEST HEADERS

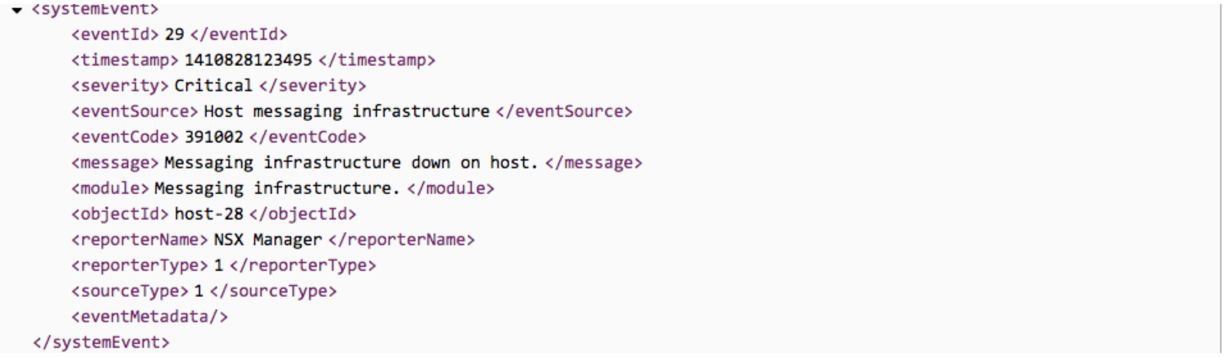
Accept: */*
Accept-Encoding: gzip, deflate, sdch
Accept-Language: en-US,en;q=0.8
Authorization: Basic YWRtaW46Y2EkaGMwdw==
Content-Type: xml
Cookie: JSESSIONID=534974A8CE0E4868...
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac O...

Note: XHR automatically adds headers like Accept, Accept-Language, Cookie, User-Agent, etc.

BODY

```
<?xml version="1.0" encoding="UTF-8" ?>
<pagedSystemEventList>
  <dataPage>
    <pagingInfo> … </pagingInfo>
    <systemEvent> … </systemEvent>
    <systemEvent> … </systemEvent>
    <systemEvent> … </systemEvent>
    <systemEvent> … </systemEvent>
    <systemEvent> … </systemEvent>
    <systemEvent> … </systemEvent>
    <systemEvent> … </systemEvent>
    <systemEvent> … </systemEvent>
    <systemEvent> … </systemEvent>
    <systemEvent> … </systemEvent>
    <systemEvent> … </systemEvent>
    <systemEvent> … </systemEvent>
    <systemEvent> … </systemEvent>
    <systemEvent> … </systemEvent>
    <systemEvent> … </systemEvent>
    <systemEvent> … </systemEvent>
    <systemEvent> … </systemEvent>
```

Multiple system events are recorded in the Response Window above. You can expand the <systemEvent> tab to see the details of a specific systemEvent. The output will look similar to:

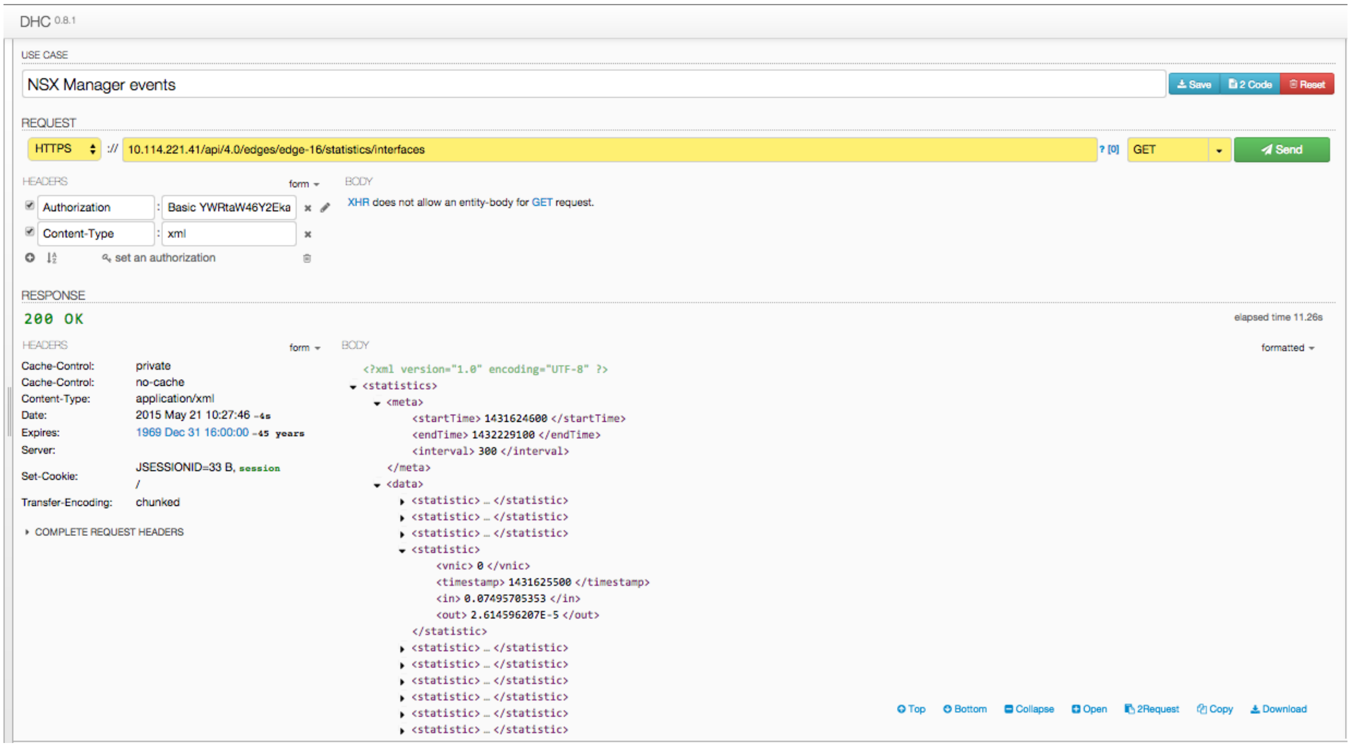


```
<systemEvent>
    <eventId> 29 </eventId>
    <timestamp> 1410828123495 </timestamp>
    <severity> Critical </severity>
    <eventSource> Host messaging infrastructure </eventSource>
    <eventCode> 391002 </eventCode>
    <message> Messaging infrastructure down on host. </message>
    <module> Messaging infrastructure. </module>
    <objectId> host-28 </objectId>
    <reporterName> NSX Manager </reporterName>
    <reporterType> 1 </reporterType>
    <sourceType> 1 </sourceType>
    <eventMetadata/>
</systemEvent>
```

NSX REST APIs can also be used for capturing statistics for monitoring purposes. Given below is an example of executing an API call to capture the interface statistics on an edge device with an ID edge-16.

API call:

```
GET https://admin:admin@10.114.221.41/api/4.0/edges/edge-16/statistics/interfaces
```



NSX Manager events

HTTPS :// 10.114.221.41/api/4.0/edges/edge-16/statistics/interfaces GET

☑ Authorization	Basic YWRtaW46Y2Eka
☑ Content-Type	xml

XHR does not allow an entity-body for GET request.

200 OK

Cache-Control:	private
Cache-Control:	no-cache
Content-Type:	application/xml
Date:	2015 May 21 10:27:46 -4s
Expires:	1969 Dec 31 16:00:00 -45 years
Server:	
Set-Cookie:	JSESSIONID=33 B, session /
Transfer-Encoding:	chunked

```
<?xml version="1.0" encoding="UTF-8" ?>
<statistics>
  <meta>
    <startTime> 1431624600 </startTime>
    <endTime> 1432229100 </endTime>
    <interval> 300 </interval>
  </meta>
  <data>
    <statistic> … </statistic>
    <statistic> … </statistic>
    <statistic> … </statistic>
    <statistic>
      <vnic> 0 </vnic>
      <timestamp> 1431625500 </timestamp>
      <in> 0.07495705353 </in>
      <out> 2.614596207E-5 </out>
    </statistic>
    <statistic> … </statistic>
    <statistic> … </statistic>
    <statistic> … </statistic>
    <statistic> … </statistic>
    <statistic> … </statistic>
    <statistic> … </statistic>
```

The response to the API call has the statistics for all the interfaces. Figure above has the statistics for vnic0 displayed from the API response.

NSX APIs are available for collecting firewall statistics, tunnel statistics, interface statistics. For a complete list of available APIs for statistics collection, please refer to the NSX-v API Guide.

Using NSX REST API with cURL

cURL is a command line tool that can be used to interact with NSX REST API using URL syntax.

Here is an example of using the NSX web service API from the command line using curl to capture all the system events.

```
# curl –insecure https://admin:admin@10.114.221.41/api/2.0/systemevent
```

Example of using the NSX web service API from the command line using curl to show the Edge status.

```
#curl --insecure https://admin:admin@10.114.221.41/api/4.0/edges/edge-16/status
```

```
<?xml version="1.0" encoding="UTF-
8"?><edgeStatus><timestamp>1431715832689</timestamp><systemStatus>good</systemStatus><activeV
seHaIndex>0</activeVseHaIndex><edgeStatus>GREEN</edgeStatus><publishStatus>APPLIED</publishSt
atus><version>84</version><edgeVmStatus/><featureStatuses><featureStatus><service>dns</servic
e><status>up</status></featureStatus><featureStatus><service>dhcp</service><status>down</stat
us></featureStatus><featureStatus><service>highAvailability</service><status>up</status></fea
tureStatus><featureStatus><service>loadBalancer</service><status>up</status></featureStatus><
featureStatus><service>nat</service><status>Applied</status></featureStatus><featureStatus><s
ervice>syslog</service><status>down</status></featureStatus><featureStatus><service>firewall<
/service><status>Applied</status></featureStatus><featureStatus><service>sslvpn</service><sta
tus>not_configured</status></featureStatus><featureStatus><service>l2vpn</service><status>not
_configured</status></featureStatus><featureStatus><service>routing</service><status>Applied<
```

```
/status></featureStatus><featureStatus><service>ipsec</service><status>not_configured</status
></featureStatus
```

You can use curl commands in a script to periodically collect the responses in a file. An example of such a script is below where the API call collects the Edge status periodically and the response of the API call is send to a file named edgestatus.txt.

```
#!/bin/bash

while true; do

curl --insecure https://admin:admin@10.114.221.41/api/4.0/edges/edge-16/status --o
edgestatus.txt

echo ""

sleep 60

done
```

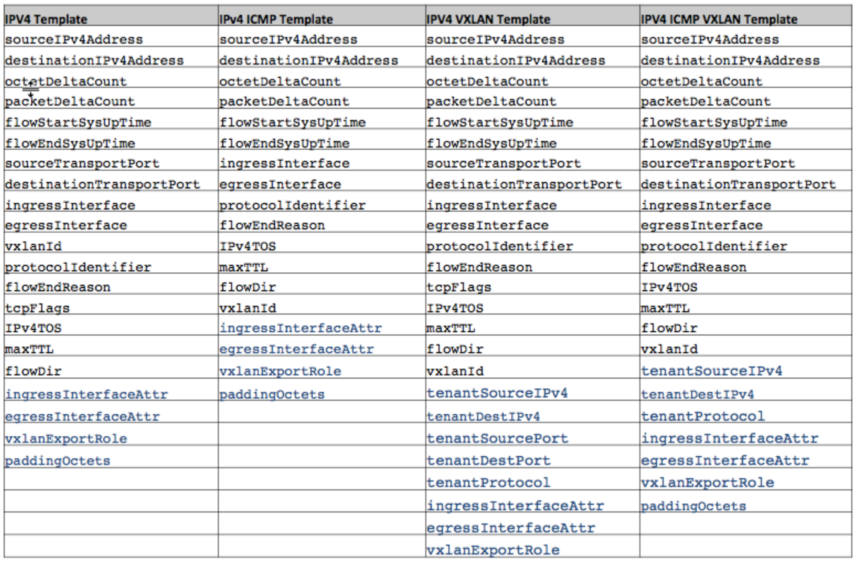
One can use a script to parse through the response file for specific data and use that data for monitoring purposes. For example, you can run the above script and within the response, filter for “><edgeStatus>GREEN” to verify the status of the Edge Appliance.

Alternatively, you can write a python script to run the API command periodically and define actions within the script for certain object values.

For more details on NSX REST APIs available for operations and monitoring, please refer to the NSX-v API Guide, available from http://pubs.vmware.com/NSX-61/index.jsp by clicking the "PDF Product Documentation" link.

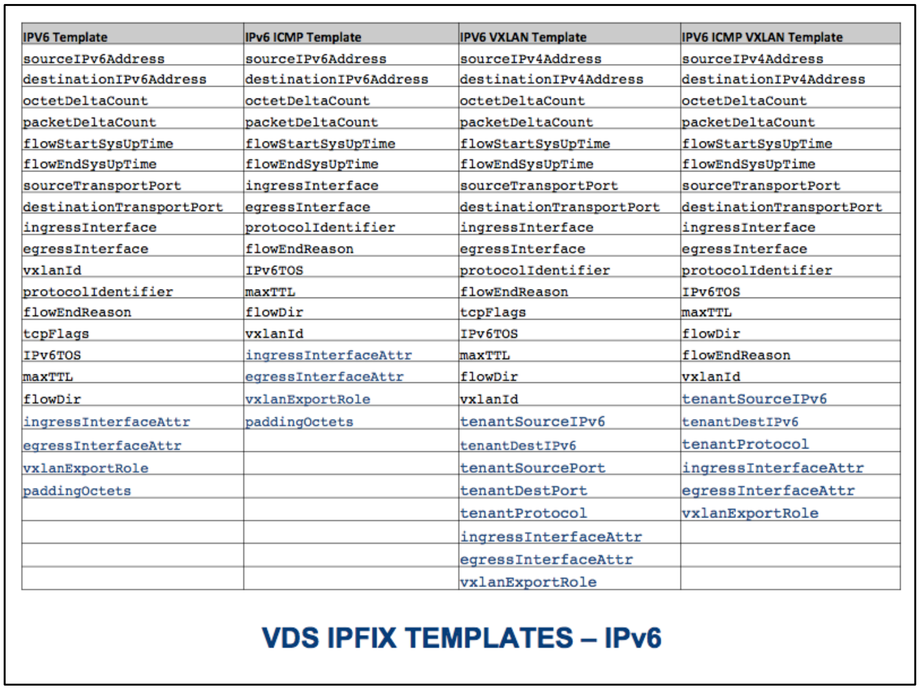
Appendix B: IPFIX Templates

VDS IPFIX Templates are shown below



IPV4 Template	IPv4 ICMP Template	IPV4 VXLAN Template	IPV4 ICMP VXLAN Template
sourceIPv4Address	sourceIPv4Address	sourceIPv4Address	sourceIPv4Address
destinationIPv4Address	destinationIPv4Address	destinationIPv4Address	destinationIPv4Address
octetDeltaCount	octetDeltaCount	octetDeltaCount	octetDeltaCount
packetDeltaCount	packetDeltaCount	packetDeltaCount	packetDeltaCount
flowStartSysUpTime	flowStartSysUpTime	flowStartSysUpTime	flowStartSysUpTime
flowEndSysUpTime	flowEndSysUpTime	flowEndSysUpTime	flowEndSysUpTime
sourceTransportPort	ingressInterface	sourceTransportPort	sourceTransportPort
destinationTransportPort	egressInterface	destinationTransportPort	destinationTransportPort
ingressInterface	protocolIdentifier	ingressInterface	ingressInterface
egressInterface	flowEndReason	egressInterface	egressInterface
vxlanId	IPv4TOS	protocolIdentifier	protocolIdentifier
protocolIdentifier	maxTTL	flowEndReason	flowEndReason
flowEndReason	flowDir	tcpFlags	IPv4TOS
tcpFlags	vxlanId	IPv4TOS	maxTTL
IPv4TOS	ingressInterfaceAttr	maxTTL	flowDir
maxTTL	egressInterfaceAttr	flowDir	vxlanId
flowDir	vxlanExportRole	vxlanId	tenantSourceIPv4
ingressInterfaceAttr	paddingOctets	tenantSourceIPv4	tenantDestIPv4
egressInterfaceAttr		tenantDestIPv4	tenantProtocol
vxlanExportRole		tenantSourcePort	ingressInterfaceAttr
paddingOctets		tenantDestPort	egressInterfaceAttr
		tenantProtocol	vxlanExportRole
		ingressInterfaceAttr	paddingOctets
		egressInterfaceAttr	
		vxlanExportRole	

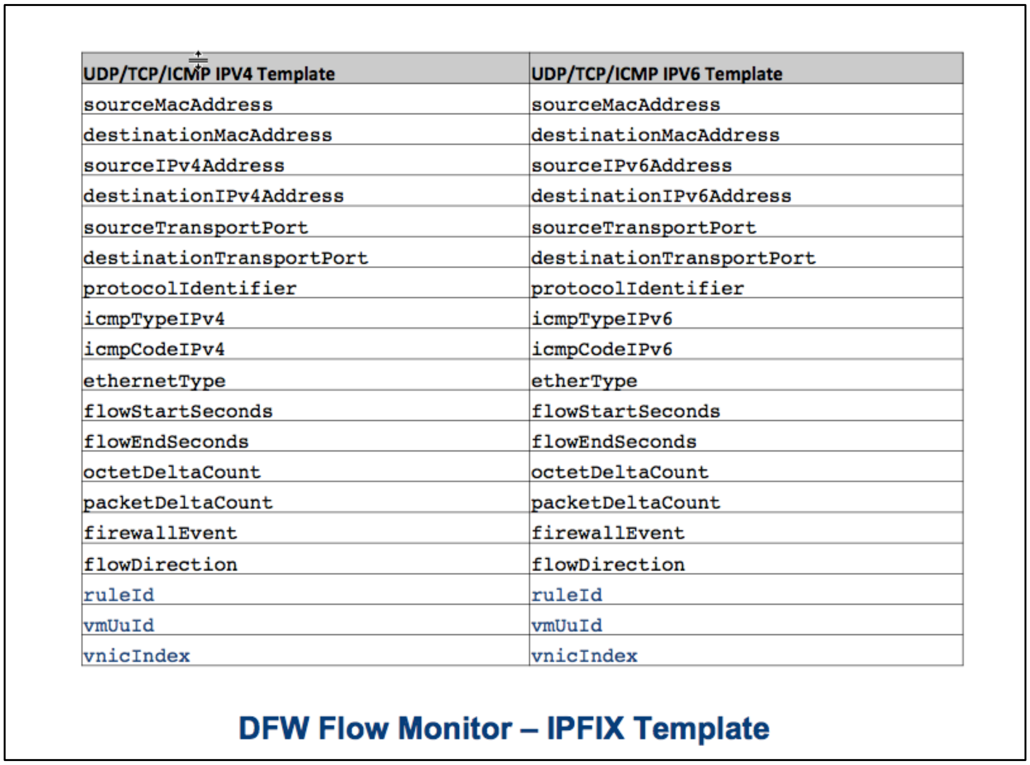
VDS IPFIX TEMPLATES – IPv4



IPV6 Template	IPv6 ICMP Template	IPV6 VXLAN Template	IPV6 ICMP VXLAN Template
sourceIPv6Address	sourceIPv6Address	sourceIPv4Address	sourceIPv4Address
destinationIPv6Address	destinationIPv6Address	destinationIPv4Address	destinationIPv4Address
octetDeltaCount	octetDeltaCount	octetDeltaCount	octetDeltaCount
packetDeltaCount	packetDeltaCount	packetDeltaCount	packetDeltaCount
flowStartSysUpTime	flowStartSysUpTime	flowStartSysUpTime	flowStartSysUpTime
flowEndSysUpTime	flowEndSysUpTime	flowEndSysUpTime	flowEndSysUpTime
sourceTransportPort	ingressInterface	sourceTransportPort	sourceTransportPort
destinationTransportPort	egressInterface	destinationTransportPort	destinationTransportPort
ingressInterface	protocolIdentifier	ingressInterface	ingressInterface
egressInterface	flowEndReason	egressInterface	egressInterface
vxlanId	IPv6TOS	protocolIdentifier	protocolIdentifier
protocolIdentifier	maxTTL	flowEndReason	IPv6TOS
flowEndReason	flowDir	tcpFlags	maxTTL
tcpFlags	vxlanId	IPv6TOS	flowDir
IPv6TOS	ingressInterfaceAttr	maxTTL	flowEndReason
maxTTL	egressInterfaceAttr	flowDir	vxlanId
flowDir	vxlanExportRole	vxlanId	tenantSourceIPv6
ingressInterfaceAttr	paddingOctets	tenantSourceIPv6	tenantDestIPv6
egressInterfaceAttr		tenantDestIPv6	tenantProtocol
vxlanExportRole		tenantSourcePort	ingressInterfaceAttr
paddingOctets		tenantDestPort	egressInterfaceAttr
		tenantProtocol	vxlanExportRole
		ingressInterfaceAttr	
		egressInterfaceAttr	
		vxlanExportRole	

VDS IPFIX TEMPLATES – IPv6

The IPFIX template for DFW Flow monitor is shown in the figure below:



UDP/TCP/ICMP IPV4 Template	UDP/TCP/ICMP IPV6 Template
sourceMacAddress	sourceMacAddress
destinationMacAddress	destinationMacAddress
sourceIPv4Address	sourceIPv6Address
destinationIPv4Address	destinationIPv6Address
sourceTransportPort	sourceTransportPort
destinationTransportPort	destinationTransportPort
protocolIdentifier	protocolIdentifier
icmpTypeIPv4	icmpTypeIPv6
icmpCodeIPv4	icmpCodeIPv6
ethernetType	etherType
flowStartSeconds	flowStartSeconds
flowEndSeconds	flowEndSeconds
octetDeltaCount	octetDeltaCount
packetDeltaCount	packetDeltaCount
firewallEvent	firewallEvent
flowDirection	flowDirection
ruleId	ruleId
vmUuId	vmUuId
vnicIndex	vnicIndex

DFW Flow Monitor – IPFIX Template

Appendix C: Log Message Codes

NSX Manager Log Messages

Appliance Name	Module	Log Message	Severity	Description
NSX Manager	vCenter	eventCode:11002	Critical	Unable co connect to vCenter server. Bad username/Password
NSX Manager	vCenter	eventCode:11005	Informational	vCenter server disconnected
NSX Manager	vCenter	eventCode:11006	Critical	Lost vCenter Server connectivity
NSX Manager	vShield Edge	eventCode:30032	High	Edge appliance not found in the vCenter directory
NSX Manager	vShield Edge	eventCode:30033	Medium	Edge VM is not responding to Healthcheck
NSX Manager	vShield Edge	eventCode:30011	Critical	None of the deployed Edge VMs is in "self" or "active" state
NSX Manager	vShield Edge	eventCode:30034	Critical	None of the NSX Edge VMs found in serving state. There is a possibility of network disruption
NSX Manager	vShield Edge	eventCode:30014	Critical	Failed to communicate with the Edge VM
NSX Manager	vShield Edge	eventCode:30013	Critical	Edge is in bad state. Need a force sync
NSX Manager	vShield Edge	eventCode:30148	Critical	NSX Edge CPU usage has increased. CPU usage: 99.06%
NSX Manager	vShield Edge	eventCode:30035	Major	Edge Communication Agent not connected to vCenter Server.
NSX Manager	vShield Edge	eventCode:30037	Critical	Edge Firewall rule modified
NSX Manager	vShield Edge	eventCode:30101	Informational	NSX Edge was booted
NSX Manager	vShield Edge	eventCode:30038	Critical	Powering on NSX Edge Appliance violates a virtual machine anti-affinity rule.
NSX Manager	vShield Edge High Availability	eventCode:30202	Major	NSX Edge HighAvailability switch over happened. VM has moved to ACTIVE state
NSX Manager	vShield Edge	eventCode:30040	Informational	NSX Edge High Availability is disabled.

Appliance Name	Module	Log Message	Severity	Description
NSX Manager	SpoofGuard	eventCode:301502	Critical	Spoofguard configuration update number 1430246410313 to host host-36 timed out
NSX Manager	vShield Firewall	eventCode:301513	Major	Firewall is uninstalled on host
NSX Manager	vShield Edge Load Balancer	eventCode:30302	Critical	Event Message:'LoadBalancer virtualServer/pool : Three-Tier-App-Members Protocol : any serverIp : web-sv-03a changed the state to down', Module:'vShield Edge LoadBalancer'
NSX Manager	Host Preparation		Critical	NSX preparation related issue (Not Ready) - EAM communication
NSX Manager	NSX Controller		Error	NSX Controller deployment issue
NSX Manager	Messaging Infrastructure	eventCode :391002 or Messaging infrastructure down on host	Critical	Messaging infrastructure down on host

NSX Controller Log Messages

Appliance Name	Log Message	Severity	Description
NSX Controller		Critical	Controller to Host communication channel failed
NSX Controller	{} closed by peer {}		Connection reset by host
NSX Controller	SSL Excpetion	Warn	SSL connection fail (handshake fail / SSL codec fail) in RX/TX
NSX Controller	SelfSignedX509TrustManager	Error	SSL certificate error/thumbprint mismatch
NSX Controller	SSLFactory	Error	Exception for SSL context
NSX Controller	Hello negotiation failed on {}	Warn	Negotiation fail / Version mismatch
NSX Controller	Failed to send hello message to {}	Error	Failed to send Hello message
NSX Controller	Invalid message header	Error	Message decode error
NSX Controller	TlvCodec	Error	TLV decode error
NSX Controller		Critical	Controller to NSX Manager

Appliance Name	Log Message	Severity	Description
			communication channel failed
NSX Controller		Warn	SSH enabled /disabled
NSX Controller		Error	Invalid certificate
NSX Controller		Error	Invalid properpties
NSX Controller	ShardingManager - No members in cluster, cluster is likely to be down	Warn	Cluster is down
NSX Controller	Pending buffer reaches the upper limit, drop {}		Memory full / Packets drop
NSX Controller	Exception	Error	Thread exit/Unhandled Exception

Hypervisor Log Messages

Appliance Name	Log Message	Severity	Description
Hypervisor	NSX manager communication channel failed	Critical	NSX manager communication channel failed
Hypervisor	Controller Communciation channel failed	Critical	Controller Communciation channel failed
Hypervisor	Can't resolve DNS name of NSX Manager to fetch VIB	Critical	Can't resolve DNS name of NSX Manager to fetch VIB

Edge and Logical Routing Log Messages

Appliance Name	Module	Error Code	Log Message	Severity	Description
vShield Manager	appliance	30102	NSX Edge is in Bad State. Needs a force sync	Critical	When VSM reads the statusCode as BAD from VSE 1.When systemStatus output in healthCheck says "bad" 2.When statusCode on config/query calls says "73000"

Appliance Name	Module	Error Code	Log Message	Severity	Description
vShield Manager	appliance	30011	None of the NSX Edge VMs found in serving state. There is a possibility of network disruption	Critical	None of the deployed vShield Edge VMs is in "self" or "active" state
vShield Manager	appliance	30013	vShield Manager found NSX Edge in bad state. Needs a force sync	Critical	When Edge-mgmt thinks that the VSE is in bad state and needs a forceSync. 1. Empty status file is received. 2. Operation times out on vSE. 3. Format of the statusFile is not as per the decided format .
vShield Manager	appliance	30014	Failed to communicate with the NSX Edge VM	Major	Failed to communicate with the Vshield Edge VM.(Unreachable)
vShield Manager	appliance	30101	NSX Edge was booted	Critical	Edge VM powerOn from inventory updates
vShield Manager	appliance	30027	NSX Edge VM is powered off	Critical	Edge VM powerOff from inventory updates
vShield Manager	appliance	30101	NSX Edge was booted	Informational	NSX Edge was booted
vShield Manager	appliance	30032	NSX Edge appliance with vmId : {0} not found	Critical	vSE VM not found in the inventory. Even the discover based on vc InstanceUuid failed
vShield Manager	healthCheck	30033	NSX Edge VM not responding to health check	Medium	An Edge Vm did not respond to consecutive 5 healthChecks (EdgeVM: NO_PULSE)

Appliance Name	Module	Error Code	Log Message	Severity	Description
vShield Manager	healthCheck	30034	None of the NSX Edge VMs found in serving state. There is a possibility of network disruption	Critical	All of the deployed edgeVms did not respond to consecutive healthChecks (Edge: NO_PULSE)
vShield Manager	vSE Communication Appliance	30035	NSX Edge Communication Agent not connected to vCenter Server	Major	VixAgent connection not connected to VC
vShield Manager	Edge Firewall	30037	Edge firewall rule modified as {0} is no longer available for {1}.', Module	Critical	FirewallRule disabled as the groupingObject got deleted
vShield Manager	vShield Manager		Directory creation failure	Major	Directory creation failure
vShield Manager	vShield Manager		Directory deletion failure	Major	Directory deletion failure
vShield Edge	appliance		CPU overloaded	Warning	CPU is overloaded beyond the threshold value 90%
vShield Edge	appliance	30100	NSX Edge was force synced	Critical	Appliance is forceSynced
vShield Edge	appliance	30102	NSX Edge is in Bad State. Needs a force sync	Critical	vShield Edge is in Bad State. Needs a force sync
vShield Edge	appliance	30150	NSX Edge process monitor detects a process failure.	Critical	vShield Edge process monitor detects a process failure.
vShield Edge	appliance	30151	NSX Edge system time is bad.	Critical	vShield Edge system time is bad
vShield Edge	appliance	30152	NSX Edge system time sync up happens	Critical	vShield Edge system time sync up happens

Appliance Name	Module	Error Code	Log Message	Severity	Description
vShield Edge	appliance	30153	AESNI crypto engine is up	Major	AESNI crypto engine is up
vShield Edge	appliance	30154	AESNI crypto engine is down	Major	AESNI crypto engine is down
vShield Edge	appliance	30180	NSX Edge is out of memory. The Edge is rebooting in 3 seconds	Critical	Out of memory, system rebooting in 3 seconds
vShield Edge	High Availability	30302	NSX Edge HighAvailability switch over happened. VM has moved to ACTIVE state	High	vShield Edge HighAvailability state is changed to ACTIVE
vShield Edge	High Availability	30303	NSX Edge HighAvailability switch over happened. VM has moved to STANDBY state	High	vShield Edge HighAvailability state is changed to STANDBY
vShield Edge	GSLB	30503	Global Loadbalancer member: {0} of pool: {1} changed the status to down	Critical	A GSLB pool member status is changed to down
vShield Edge	GSLB	30506	Global Loadbalancer peer {0} of site: {1} changed the status to down	Critical	A GSLB peer site status is changed to down
vShield Edge	OSPF		Packet received with invalid area ID	Exception	
vShield Edge	OSPF		An adjacency with a neighbor has gone down	Audit	

Logical Switch Log Messages

Log Message	Severity	Description
Vdn Scope: {} not found	Error	Failed to create Logical Switch due to invalid transport zone
Controller not present or not supported on backing switches	Error	Invalid control plane mode selected if the underlying DVS is at lower version (<5.5) or if controllers not present for unicast/hybrid mode
Invalid Control Plane Mode Selected for {}. Some clusters are not controller ready.	Error	Invalid control plane mode if some clusters are not controller ready
IPPool {} does not have sufficient resources.	Error	VTEP creation failed due to IP pool exhaustion
VirtualWireServiceImpl:878 - Could not create backing for virtual wire	Error	Could not create backing for virtual wire (Module : VirtualWire, bug : 1306259)
MAC record {} to remove by {} not exists	Warn	Remove non-existing MAC record
MAC record {} to add by {} already exists	Warn	Add duplicate MAC record
ARP record {} to add by {} already exists	Warn	Add duplicate ARP record
ARP record {} to remove by {} not exists	Warn	Remove non-existing ARP record
Conflicted VTEP {} added by {}	Warn	Add conflicted VTEP
error {} to add by {} already exists	Warn	Add duplicate VTEP record
VTEP {} to remove by {} not exists	Warn	Remove non-existing VTEP
Change not-existed VTEP {} by {}	Warn	Change non-existing VTEP
Try to join not existed VNI {} by {}	Warn	Controller refusing to serve VXLAN instance assigned to it
Leave not existed VNI {} by {}	Warn	Leave non-existing VNI
{} tries to leave not joined VNI {}	Warn	Try to leave not joined VNI
Update existing VNI {}	Error	Remove non-existing VNI from mgmt
Request not existed VNI {} by {}	Warn	Process Event on closed connection
Request not existed VNI {} by {}	Warn	Request non-existing VNI
Exception when register JMX MBean	Error	Exception when register JMX MBean
Change from {} to a duplicated VTEP record {}	Error	Active VTEP IP changed from a duplicated one
Received invalid * message from {}, * should be any of query, update, flush, notification	Error	Message decode error or invalid message
Received not supported message {} from	Error	Unsupported message

Log Message	Severity	Description
{}		
Failed to post event for {} from {}	Error	Failed to post a mapping event to worker thread
InterruptedException while process {}	Error	Failed to handle cluster down event because of interrupted exception
{} already closed or not opened yet	Error	Failed to handle CLI request because of connection closed
Failed to handle cli event {}	Error	Failed to handle CLI request because of result is null
Exception or error in vxlan worker	Error	Unhandled error in worker thread
WARNING: vxlan: VDL2PortPropSet:170: Failed to create VXLAN IP vmknic on port[0x400000b] of VDS[DvsPortset-2] : Failure	Critical	Failed to create VTEP interface - likely due to VSM negotiating wrong VMODL version with VC
WARNING: vxlan: VDL2PortPropSet:170: Failed to create VXLAN IP vmknic on port[0x300000e] of VDS[DvsPortset-0] : Would block	Critical	VXLAN configuration pushed to host before host was prep'ed - host must be rebooted to initialize configuration in correct order
WARNING: vxlan: VDL2Init:558: Failed to find netstack 'vxlan'	Critical	VXLAN TCP/IP stack not created - On stateless hosts this indicates incorrect host profile creation steps
WARNING: vxlan: VDL2CPCheckConnUpCB:311: Control plane connection of VXLAN network[5007] is down vxlan: VDL2CPProcessLinkChange:3135: Control plane link down[IP: 7518240a] for VNI[5000]	Error	VXLAN dataplane lost connection to controller
WARNING: vxlan: VDL2CheckVmknicStatus:987:No valid VDL2 IP for a long time in VLAN[4090] on VDS[DvsPortset-1]	Critical	VTEP does not have valid IP address

Host Preparation Log Messages

Log Message	Severity	Description
Error encountered in invoking handler com.vmware.vshield.vsm.common.messaginginfra.UserWorldSetupHandler$HostPreparedEventHandler for event HostPreparedEvent	Error	MessageBus (RMQ channel) setup fails after Host was Prepared

Log Message	Severity	Description
ClientServiceImpl:280 - Messaging system credentials verification failed	Error	MessageBus (RMQ channel) setup fails after Host was Prepared/ RePrepared/ ReConnected due to Creds Verification failure
Error encountered in invoking handler com.vmware.vshield.vsm.common.messaginginfra.UserWorldSetupHandler$PreparedHostConnectedEventHandler for event PreparedHostConnectedEvent. com.vmware.vshield.vsm.inventory.vcoperations.VcOperationFailedException: core-services:1500:Operation failed on VC. For more details, refer to the rootCauseString or the VC logs:'UserVars.RmqClientToken' is invalid or exceeds the maximum number of characters permitted.	Error	Setting up of Advance Configs failed on Host, when Prepared Host re-connected. This happens mainly when VSM pushes AdvConfigs before VIBs were fully installed.
Error encountered in invoking handler com.vmware.vshield.xvs.drivers.vc.service.VdnHostInstallationServiceImpl$HostRePreparedHandler for event HostRePreparedEvent	Error	VXLAN realted HostRePrepHandler failed, this event handler is invoked when EAM gave GREEN for already Prepared Host
Severity:'Critical', Event Source:'VXLAN Kernel Module', Code:'1902', Event Message:'VXLAN instance does not exist on the host 'host-31'.', Module:'VXLAN'	Error	VxLAN not setup correctly on the host. If the vib is installed on the host after the vxlan properties are pushed, then the host ends up in a bad state. A subsequent reboot fixes it

DFW Firewall Log Messages

Appliance Name	Module	Error Code	Log Message	Severity	Description
vShield Manager	FwConfig	301501	Firewall_CONFIG_UPDATE_TIMEDOUT	Critical	This is vsm side event if host failed to respond with in time out window.
vShield Manager	SpoofGuard	301502	SPOOFGUARD_CONFIG_UPDATE_TIMEDOUT	Critical	This is vsm side event if host failed to respond with in time out window.
vShield Manager	FwConfig	301503	FIREWALL_CONFIG_PREPARE_FAILED	Critical	This is vsm side event if vsm failed while provisioning firewall rule
vShield Manager	SGconfig	301504	CONTAINER_CONFIG_PREPARE_FAILED	Critical	This is vsm side event if vsm failed to send container update
vShield Manager	SpoofGuard	301505	SPOOFGUARD_CONFIG_PREPARE_FAILED	Critical	This is vsm side event if vsm failed to send container update
vShield Manager	Firewall	301506	FILTER_SET_PREPARE_FAILED	Critical	This is vsm side event if vsm failed to send exclude list update

Appliance Name	Module	Error Code	Log Message	Severity	Description
vShield Manager	Firewall	301507	FIREWALL_RULE_AFFECTED	Information	On deletion of grouping object if firewall rule is affected then event is raised with description listing affected rule Ids.
UW	FilterConfig	301001	FILTER_CONFIG_UPDATE_FAILED	Critical	Host failed to receive/parse filter config or open device /dev/dvfiltertbl. Key value pair would have possible cause of failure and context information.
UW	FilterConfig	301002	FILTER_CONFIG_NOT_APPLIED	Major	Failed to apply Filter config to vnic. Key value pair would have possible cause of failure (like failed while opening/parsing/updating filter config) and context information.
UW	SPconfig	301011	SERVICE_PROFILE_UPDATE_FAILED	Critical	Host failed to parse service profile config or open vsip device. Key value pair would have possible cause of failure (like failed while opening/parsing/updating filter config) and context information.
UW	SPconfig	301012	SERVICE_CONFIG_NOT_APPLIED	Major	Failed to update service profile config. Key value will have context info like profile name / version.
UW	SPconfig	301015	SERVICE_CONFIG_DELETE_FAILED	Major	Failed to delete service profile config. Key value will have context info like profile name / version.
UW	SPconfig	301015	SERVICE_CONFIG_DELETED	Information	Deleted service profile config. Key value will have context info like profile name / version
UW	FwConfig	301031	Firewall_CONFIG_UPDATE_FAILED	Critical	Failed to receive/parse/Update firewall config. Key value will have context info like generation number and also other debugging info.
UW	FwConfig	301033	Firewall_RULE_NOT_APPLIED	Major	Failed to apply firewall config. Key value will have context info like generation number and also other debugging info.
UW	SGconfig	301041	CONTAINER_CONFIG_UPDATE_FAILED	Critical	Failed receive/parse/update container config. Key value will have context info like container name & generation number.
UW	Flow	301051	FLOW_MISSED	Major	Flow missed. Key value will have details like count and debugging info like flow dropped for filter or failed to read flow from filter or failed to send flow to vsm.

Appliance Name	Module	Error Code	Log Message	Severity	Description
UW	SpoofGuard	301061	SpoofGuard_CONFIG_ UPDATE_FAILED	Critical	Failed to receive/parse/Update spoofguard config. Key value will have context info like generation number and also other debugging info.
UW	SpoofGuard	301062	SpoofGuard_CONFIG_ APPLIED_FAILED	Major	Failed to apply spoofguard for vnic.
UW	SpoofGuard	301064	SpoofGuard_CONFIG_ DELETE_FAILED	Major	Failed to disable spoofguard for vnic.
UW	SpoofGuard	301065	SpoofGuard_CONFIG_ DELETED	Information	Disabled spoofguard for vnic.
UW	VSFWD	301071	VSFWD_STARTED	Information	Whenver vsfwd starts event will be raise.