DELL EMC VXFLEX OS
Networking Best Practices and Design Considerations

ABSTRACT
This document describes core concepts and best practices for designing, troubleshooting, and maintaining a Dell EMC™ VxFlex OS® network of any size.

July 2018
# TABLE OF CONTENTS

**EXECUTIVE SUMMARY** ........................................................................................................... 5
**AUDIENCE AND USAGE** .......................................................................................................... 5
**VXFLEX OS FUNCTIONAL OVERVIEW** .................................................................................... 6

**VXFLEX OS SOFTWARE COMPONENTS** .................................................................................... 7
- Storage Data Servers (SDS) .......................................................................................................... 7
- Storage Data Clients (SDC) .......................................................................................................... 8
- Meta Data Managers (MDM) ....................................................................................................... 8

**TRAFFIC TYPES** ....................................................................................................................... 8
- Storage Data Client (SDC) to Storage Data Server (SDS) ........................................................... 9
- Storage Data Server (SDS) to Storage Data Server (SDS) .......................................................... 9
- Meta Data Manager (MDM) to Meta Data Manager (MDM) ..................................................... 9
- Meta Data Manager (MDM) to Storage Data Client (SDC) ....................................................... 9
- Meta Data Manager (MDM) to Storage Data Server (SDS) ...................................................... 9
- Other Traffic ................................................................................................................................ 9

**NETWORK INFRASTRUCTURE** .................................................................................................. 10
- Flat Network Topologies ............................................................................................................. 11
- IPv4 and IPv6 .............................................................................................................................. 12

**NETWORK PERFORMANCE AND SIZING** ............................................................................ 12
- Network Latency ....................................................................................................................... 12
- Network Throughput ................................................................................................................. 12
  - Example: An SDS-only node with 10 HDDs ......................................................................... 12
  - Example: An SDS-only node with 6 SSDs and 18 HDDs ..................................................... 13
- Write-heavy environments ......................................................................................................... 13
- Hyper-converged environments ............................................................................................... 13

**NETWORK HARDWARE** .......................................................................................................... 14
- Dedicated NICs ........................................................................................................................... 14
- Two NICs vs. Four NICs and Other Configurations ................................................................... 14
- Switch Redundancy ................................................................................................................... 14
- Buffer Capacity ......................................................................................................................... 14

**IP CONSIDERATIONS** ............................................................................................................. 15
- IP-level Redundancy .................................................................................................................. 15
Ingress Replication .................................................................................................................. 27
Multicast ................................................................................................................................ 28
VXLAN with vPC ....................................................................................................................... 29
Other Hardware based VXLAN Considerations .................................................................. 30
Software based VXLAN with VMware NSX ........................................................................ 30
Software Considerations with NSX ...................................................................................... 32
Hardware and Performance Considerations with VMware NSX ......................................... 32

VALIDATION METHODS ........................................................................................................ 32
VxFlex OS Native Tools .......................................................................................................... 32
   SDS Network Test ................................................................................................................ 32
   SDS Network Latency Meter Test ....................................................................................... 33
Iperf, NetPerf, and Tracepath .................................................................................................. 34
Network Monitoring ................................................................................................................ 34
Network Troubleshooting Basics ........................................................................................... 34

SUMMARY OF RECOMMENDATIONS ..................................................................................... 35
Traffic Types ............................................................................................................................. 35
Network Infrastructure ............................................................................................................. 35
Network Performance and Sizing ............................................................................................ 35
Network Hardware ................................................................................................................... 35
IP Considerations ..................................................................................................................... 36
Ethernet Considerations ......................................................................................................... 36
Link Aggregation Groups ......................................................................................................... 36
The MDM Network ................................................................................................................ 36
Network Services ..................................................................................................................... 36
Dynamic Routing Considerations ........................................................................................... 36
Hardware-based VXLAN Considerations .............................................................................. 37
Software-based VXLAN Considerations ................................................................................. 37
VMware Considerations ......................................................................................................... 38
Validation Methods ................................................................................................................ 38

CONCLUSION .......................................................................................................................... 38
EXECUTIVE SUMMARY

The Dell EMC™ Flex family of products is powered by VxFlex OS – a scale-out block storage service that enables customers to create a resilient server SAN or hyper-converged infrastructure on x86 server hardware. The Flex family accommodates a wide variety of deployment options, with multiple OS and hypervisor capabilities, and is ideal for applications requiring high performance and ease of management as one scales from small to large. The Flex family currently consists of a rack-level and a node-level offering: Dell EMC VxRack FLEX and the Dell EMC VxFlex Ready Node. This document is primarily relevant to the VxFlex Ready Nodes, but will be of interest to anyone wishing to understand the networking required for a successful VxFlex OS-based storage system.

Dell EMC VxRack FLEX, on the other hand, is a fully engineered, rack-scale hyper-converged system that delivers flexibility, scalability and performance for the enterprise data center. It is a quick and easy to deploy engineered solution in which the networking is already configured and optimized. For other Flex family solutions, one must design and implement an appropriate network.

VxFlex OS is used to build robust, enterprise storage systems, and one of the key architectural advantages of VxFlex OS is that it distributes load evenly and symmetrically across all contributing server nodes. This eliminates concerns associated with bottlenecks at storage protocol endpoints. It also frees storage administrators from micromanagement by moving the granularity of operations away from individual components to the clustered infrastructure. Because networks are the central component of data center infrastructure, understanding their relationship to VxFlex OS is crucial for a successful deployment.

A successful VxFlex Ready Node deployment depends on a properly designed network topology. This guide provides details on network topology choices, network performance, hyper-converged considerations, Ethernet considerations, dynamic IP routing considerations, VxFlex OS implementations within a VMware® environment, validation methods, and monitoring recommendations.

Audience and Usage

This document is meant to be accessible to readers who are not networking experts. However, an intermediate level understanding of modern IP networking is assumed.

The “Dynamic Routing Considerations” and “VXLAN” sections of this document contain information and concepts that are likely unfamiliar to storage and virtualization administrators. However, this white paper was written with storage and virtualization administrators in mind, so those concepts are described methodically and with care.

Readers familiar with VxFlex OS may choose to skip the “VxFlex OS Functional Overview” and “VxFlex OS Software Components” sections. Specific recommendations appearing in bold are re-visited in the “Summary of Recommendations” section near the end of this document.

This guide provides a minimal set of network best practices. It does not cover every networking best practice for VxFlex OS. A VxFlex OS technical expert may recommend more comprehensive best practices than covered in this guide.

Cisco Nexus® switches are used in the examples in this document, but the same principles will apply to any network vendor. For detailed guidance in the use of Dell network equipment, see the ScaleIO/VxFlex OS IP Fabric Best Practice and Deployment Guide with Dell EMC Networking OS10 Enterprise Edition.
VxFlex OS Functional Overview

VxFlex OS is software that creates a server and IP-based SAN from direct-attached server storage to deliver flexible and scalable performance and capacity on demand. As an alternative to a traditional SAN infrastructure, VxFlex OS combines HDDs, SSDs, and PCIe flash cards to create a virtual pool of block storage with varying performance tiers. VxFlex OS provides enterprise-grade data protection, multi-tenant capabilities, and add-on enterprise features such as QoS, thin provisioning, and snapshots. VxFlex OS supports physical and virtual application servers, and has been proven to deliver significant TCO savings vs. traditional SAN. VxFlex OS provides the following benefits:

Massive Scale - VxFlex OS can scale from three to 1024 nodes. The scalability of performance is linear with regard to the growth of the deployment. As devices or nodes are added, VxFlex OS automatically redistributes data evenly, resulting in a fully balanced pool of distributed storage.

Extreme Performance - Every device in a VxFlex OS storage pool is used to process I/O operations. This massive I/O parallelism of resources eliminates bottlenecks. Throughput and IOPS scale in proportion to the number of storage devices added to the storage pool. Performance and data protection optimization is automatic. Component loss triggers a rebuild operation to preserve data protection. Addition of a component triggers a rebalance to increase available performance and capacity. Both operations occur in the background with no downtime to applications and users.

Compelling Economics - VxFlex OS does not require a Fibre Channel fabric or dedicated components like HBAs. There are no forklift upgrades for outdated hardware. Failed and outdated components are simply removed from the system. VxFlex OS can reduce the cost and complexity of the solution and has been proven to result in significant TCO savings vs. traditional SAN.¹

Unparalleled Flexibility - VxFlex OS provides flexible deployment options. In a two-layer deployment, applications and the storage software are installed on separate pools of servers. A two-layer deployment allows compute and storage teams to maintain operational autonomy. In a hyper-converged deployment, applications and storage are installed on a shared pool of servers. This provides the lowest footprint and cost profile. The deployment models can also be mixed to provide independent scaling of compute and storage resources.

Supreme Elasticity - Storage and compute resources can be increased or decreased whenever the need arises. The system automatically rebalances data on the fly. Additions and removals can be done in small or large increments. No capacity planning or complex reconfiguration is required. Rebuild and rebalance operations happen automatically without operator intervention.

Essential Features for Enterprises and Service Providers - With VxFlex OS, you can limit the amount of performance (IOPS or bandwidth) that selected customers can consume. Quality of Service allows for resource usage to be dynamically managed, addressing any bully workload scenarios. VxFlex OS also offers instantaneous, writeable snapshots for data backups and cloning.

DRAM caching enables you to improve read performance by using server RAM. Any group of servers hosting storage that may fail together (such as nodes in the same rack) can be grouped together in a fault set. Fault sets can be defined to ensure data mirroring occurs outside the failure group, improving business continuity. Volumes can be thin provisioned, providing on-demand storage as well as faster setup and startup times.

VxFlex OS also provides multi-tenant capabilities via protection domains and storage pools. Protection Domains allow you to isolate specific servers and data sets. Storage Pools can be used for further data segregation, tiering, and performance management. For example, data that is accessed very frequently can be stored in a flash-only storage pool for the lowest latency, while less frequently accessed data can be stored in a low-cost, high-capacity pool of spinning disks.

VxFlex OS Software Components

VxFlex OS fundamentally consists of three types of software components: the Storage Data Server (SDS), the Storage Data Client (SDC), and the Meta Data Manager (MDM).

A logical illustration of a VxFlex OS deployment. Each volume available to an SDC is distributed across many systems running the SDS. The Meta Data Managers (MDMs) reside outside the data path, and they are only consulted by SDCs when an SDS fails or when the data layout changes.

STORAGE DATA SERVERS (SDS)

The Storage Data Server (SDS) aggregates and serves raw local storage in a server as part of a VxFlex OS cluster. The SDS is the server-side software component. A server that takes part in serving data to other nodes has an SDS service installed and running on it. A collection of SDSs form the VxFlex OS persistence layer.

SDSs maintain redundant copies of the user data, protect each other from hardware loss, and reconstruct data protection when hardware components fail. SDSs may leverage SSDs, PCIe based flash, spinning media, RAID controller write caches, available RAM, or any combination thereof.

SDSs may run natively on Windows or Linux, or as a virtual appliance on ESX. A VxFlex OS cluster may have 1024 nodes, each running an SDS. Each SDS requires only 500 megabytes of RAM.

SDS components can communicate directly with each other. The SDS components are fully meshed. And SDSs are optimized for rebuild, rebalance, and I/O parallelism. Data layout between SDS components is managed through storage pools, protection domains, and fault sets.

Client volumes used by the SDCs are placed inside a storage pool. Storage pools are used to logically aggregate types of storage media at drive-level granularity. Storage pools provide varying levels of storage service priced by capacity and performance.

Protection from node, device, and network connectivity failure is managed with node-level granularity through protection domains. Protection domains are groups of SDSs where replicas are maintained.

Fault sets allow large systems to tolerate multiple simultaneous failures by preventing redundant copies from residing in a single node, rack or chassis.
STORAGE DATA CLIENTS (SDC)

The Storage Data Client (SDC) allows an operating system or hypervisor to access data served by VxFlex OS clusters. The SDC is a client-side software component that can run natively on Windows®, Linux, IBM AIX®, or ESX®. It is analogous to a software initiator, but is optimized to use multiple networks and endpoints in parallel.

The SDC provides the operating system or hypervisor running it with access to logical block devices called “volumes”. A volume is analogous to a LUN in a traditional SAN. Each logical block device provides raw storage for a database or a file system.

The SDC knows which Storage Data Server (SDS) endpoints to contact based on block locations in a volume. The SDC consumes distributed storage resources directly from other systems running VxFlex OS. SDCs do not share a single protocol target or network endpoint with other SDCs. SDCs distribute load evenly and autonomously.

The SDC is extremely lightweight. SDC to SDS communication is inherently multi-pathed across SDS storage servers, in contrast to approaches like iSCSI, where multiple clients target a single protocol endpoint. This enables much better performance scalability.

The SDC allows shared volume access for uses such as clustering. The SDC does not require an iSCSI initiator, a fibre channel initiator, or an FCoE initiator. Each SDC requires only 50 megabytes of RAM. The SDC is optimized for simplicity, speed, and efficiency.

META DATA MANAGERS (MDM)

MDMs control the behavior of the VxFlex OS system. They determine and publish the mapping between clients and their data, keep track of the state of the system, and issue reconstruct directives to SDS components.

MDMs establish the notion of quorum in VxFlex OS. They are the only tightly clustered component of VxFlex OS. They are authoritative, redundant, and highly available. They are not consulted during I/O operations or during SDS to SDS operations like rebuild and rebalance. When a hardware component fails, the MDM cluster will instruct an auto-healing operation to begin within seconds. An MDM cluster is comprised of at least three servers, or five can be used to further improve availability.

Traffic Types

VxFlex OS performance, scalability, and security can benefit when the network architecture reflects VxFlex OS traffic patterns. This is particularly true in large VxFlex OS deployments. The software components that make up VxFlex OS (the SDCs, SDSs, and MDMs) converse with each other in a predictable way. Architects designing a VxFlex OS deployment should be aware of these traffic patterns in order to make informed choices about the network layout.

A simplified illustration of how the VxFlex OS software components communicate. A VxFlex OS system will have many SDCs, SDSs, and MDMs. This illustration groups SDCs, SDSs, and MDMs. The arrows from the SDSs and MDMs pointing back to themselves represent communication to other SDSs and MDMs. The traffic patterns are the same regardless of the physical location of an SDC, SDS, or MDM.
Storage Data Client (SDC) to Storage Data Server (SDS)
Traffic between the SDCs and the SDSs forms the bulk of front end storage traffic. Front end storage traffic includes all read and write traffic arriving at or originating from a client. This network has a high throughput requirement.

Storage Data Server (SDS) to Storage Data Server (SDS)
Traffic between SDSs forms the bulk of back end storage traffic. Back end storage traffic includes writes that are mirrored between SDSs, rebalance traffic, and rebuild traffic. This network has a high throughput requirement.

Although not required, there may be situations where isolating front-end and back-end traffic for the storage network may be ideal. This is required in two-layer deployments where the storage and server teams act independently.

Meta Data Manager (MDM) to Meta Data Manager (MDM)
MDMs are used to coordinate operations inside the cluster. They issue directives to VxFlex OS to rebalance, rebuild, and redirect traffic. MDMs are redundant, and must communicate with each other to maintain a shared understanding of data layout. MDMs also establish the notion of quorum in VxFlex OS.

MDMs do not carry or directly interfere with I/O traffic. MDMs do not require the same level of throughput required for SDS or SDC traffic. **MDM to MDM traffic requires a stable, reliable, low latency network.** MDM to MDM traffic is considered back end storage traffic.

VxFlex OS supports the use of one or more networks dedicated to traffic between MDMs. At least two 10-gigabit links should be used per MDM.

Meta Data Manager (MDM) to Storage Data Client (SDC)
The master MDM must communicate with SDCs in the event that data layout changes. This can occur because the SDSs that host storage for the SDCs are added, removed, placed in maintenance mode, or go offline. Communication between the Master MDM and the SDCs is asynchronous.

**MDM to SDC traffic requires a reliable, low latency network. MDM to SDC traffic is considered front end storage traffic.**

Meta Data Manager (MDM) to Storage Data Server (SDS)
The master MDM must communicate with SDSs to issue rebalance and rebuild directives. MDM to SDS traffic requires a reliable, low latency network. **MDM to SDS traffic is considered back end storage traffic.**

Other Traffic
Other traffic includes management, installation, and reporting. This includes traffic to the VxFlex OS Gateway (REST Gateway, Installation Manager, and SNMP trap sender), traffic to and from the Light Installation Agent (LIA), and reporting or management traffic to the MDMs (such as syslog for reporting and LDAP for administrator authentication) or to an AMS (Automated Management Services) installation. See the VxFlex OS User Guide in the VxFlex OS Documentation Hub for more information.

SDCs do not communicate with other SDCs. This can be enforced using private VLANs and network firewalls.
A simple VxFlex OS IP network layout. The management network is routed, and provides access to the iDRAC, ESX, and the Storage Virtual Machine (SVM). Redundant networks carry SDS, MDM, and SDC traffic. The SDS and MDM traffic use the same set of IP addresses. The traffic is not segmented into front-end (SDS, SDC, MDM) and back-end traffic (SDS, MDM), as might be the case in a larger deployment. The 192.168.160.X and 172.17.160.X address spaces can be used for the MDM virtual IP.

**Network Infrastructure**

Leaf-spine (also called Clos) and flat network topologies are the most common in use today. Flat networks are used in very small networks. In modern datacenters leaf-spine topologies are preferred over legacy hierarchical topologies. This section compares flat and leaf-spine topologies as a transport medium for VxFlex OS data traffic.

**Dell EMC recommends the use of a non-blocking network design.** Non-blocking network designs allow the use of all switch ports concurrently, without blocking some of the network ports to prevent message loops. Therefore, Dell EMC strongly recommends against the use of Spanning Tree Protocol (STP) on a network hosting VxFlex OS.

In order to achieve maximum performance and predictable quality of service, the network should not be over-subscribed.

**Leaf-Spine Network Topologies**

A two-tier leaf-spine topology provides a single switch hop between leaf switches and provides a large amount of bandwidth between end points. A properly sized leaf-spine topology eliminates oversubscription of uplink ports. Very large datacenters may use a three-tier leaf-spine topology. For simplicity, this paper focuses on two tier leaf-spine deployments.

In a leaf-spine topology, each leaf switch is attached to all spine switches. Leaf switches do not need to be directly connected to other leaf switches. Spine switches do not need to be directly connected to other spine switches.

In most instances, Dell EMC recommends using a leaf-spine network topology. This is because:

- VxFlex OS can scale out to 1024 nodes.
- Leaf-spine architectures are future proof. They facilitate scale-out deployments without having to re-architect the network.
- A leaf-spine topology allows the use of all network links concurrently. Legacy hierarchical topologies must employ technologies like Spanning Tree Protocol (STP), which blocks some ports to prevent loops.
- Properly sized leaf-spine topologies provide more predictable latency due to the elimination of uplink oversubscription.
A two-tier leaf-spine network topology. Each leaf switch has multiple paths to every other leaf switch. All links are active. This provides increased throughput between devices on the network. Leaf switches may be connected to each other for use with MLAG (not shown).

Flat Network Topologies

A flat network topology may be easier to implement, and may be the preferred choice if an existing flat network is being extended or if the network is not expected to scale. In a flat network, all the switches are used to connect hosts. There are no spine switches.

If you expand beyond a small number of switches, the additional cross-link ports required would likely make a flat network topology cost-prohibitive.

The primary use-cases for a flat network topology are:

- Small datacenter deployments that will not grow.
- Remote office or back office.
- Small business.
IPv4 and IPv6
VxFlex OS 2.0 and greater provides IPv6 support. Earlier versions supported Internet Protocol version 4 (IPv4) addressing only.

Network Performance and Sizing
A properly sized network frees network and storage administrators from concerns over individual ports or links becoming performance or operational bottlenecks. The management of networks instead of endpoint hot-spots is a key architectural advantage of VxFlex OS.

Because VxFlex OS distributes I/O across multiple points in a network, network performance must be sized appropriately.

Network Latency
Network latency is important to account for when designing your network. Minimizing the amount of network latency will provide for improved performance and reliability. For best performance, latency for all SDS and MDM communication should not exceed 1 millisecond network-only round-trip time under normal operating conditions. Since wide-area networks’ (WANs) lowest response times generally exceed this limit, you should not operate VxFlex OS clusters across a WAN.

Latency should be tested in both directions between all components. This can be verified by pinging, and more extensively by the SDS Network Latency Meter Test. The open source tool iPerf can be used to verify bandwidth. Please note that iPerf is not supported by Dell EMC. iPerf and other tools used for validating a VxFlex OS deployment are covered in detail in the “Validation Methods” section of this document.

Network Throughput
Network throughput is also a critical component when designing your VxFlex OS implementation. Throughput is important to reduce the amount of time it takes for a failed node to rebuild; to reduce the amount of time it takes to redistribute data in the event of uneven data distribution; to optimize the amount of I/O a node is capable of delivering; and to meet performance expectations.

While VxFlex OS can be deployed on a 1-gigabit network, storage performance will likely be bottlenecked by network capacity. At a minimum, Dell EMC recommends leveraging 10-gigabit network technology.

Additionally, although the VxFlex OS cluster itself may be heterogeneous, the SDS components that make up a protection domain should reside on hardware with equivalent storage and network performance. This is because the total bandwidth of the protection domain will be limited by the weakest link during reconstruct and I/O operations.

In addition to throughput considerations, it is recommended that each node have at least two separate network connections for redundancy, regardless of throughput requirements. This remains important even as network technology improves. For instance, replacing two 10-gigabit links with a single 25- or 40-gigabit link improves throughput but sacrifices link-level network redundancy.

In most cases, the amount of network throughput to a node should match or exceed the combined maximum throughput of the storage media hosted on the node. Stated differently, a node’s network requirements are proportional to the total performance of its underlying storage media.

When determining the amount of network throughput required, keep in mind that modern media performance is typically measured in megabytes per second, but modern network links are typically measured in gigabits per second.

To translate megabytes per second to gigabits per second, first multiply megabytes by 8 to translate to megabits, and then divide megabits by 1,000 to find gigabits.

\[ \text{gigabits} = \frac{\text{megabytes} \times 8}{1,000} \]

Note that this is not precise, as it does not account for the base-2 definition of “kilo” as 1024, but it is adequate for this purpose.

Example: An SDS-only node with 10 HDDs
Assume that you have a node hosting only an SDS. This is not a hyper-converged environment, so only storage traffic must be taken into account. The node contains 10 hard disk drives. Each of these drives is individually capable of delivering a raw throughput of 100 megabytes per second under the best conditions (sequential I/O, which VxFlex OS is optimized for during reconstruct and rebalance operations). The total throughput of the underlying storage media is therefore 1000 megabytes per second.
10 \times 100 \text{ megabytes} = 1,000 \text{ megabytes}

Then convert 1,000 megabytes to gigabits using the equation described earlier, first multiply 1000MB by 8, and then divide by 1,000.

\[
\frac{1,000 \text{ megabytes} \times 8}{1,000} = 8 \text{ gigabits}
\]

In this case, if all the drives on the node are serving read operations at the maximum speed possible, the total network throughput required would be 8 gigabits per second. We are accounting for read operations only, which is typically enough to estimate the network bandwidth requirement. This can be serviced by a single 10 gigabit link. However, since network redundancy is encouraged, this node should have at least two 10 gigabit links.

**Example: An SDS-only node with 6 SSDs and 18 HDDs**

This is another two-layer example, where only storage traffic must be taken into account. In this case, the node hosts 6 SSDs that can each deliver 450 megabytes per second, and 18 HDDs that can each deliver 100 megabytes per second.

\[
(6 \times 450 \text{ megabytes}) + (18 \times 100 \text{ megabytes}) = 4,500 \text{ megabytes}
\]

The SDS has 4,500 megabytes of potential raw storage throughput. Convert the result into gigabits.

\[
\frac{4,500 \text{ megabytes} \times 8}{1000} = 36 \text{ gigabits}
\]

Four 10 gigabit links can service the node’s potential read throughput. This estimation does not account for writes, but is sufficient for most cases. Because there will be four links, the loss of a network link will not bring down this SDS, if it is configured properly.

Note that this level of throughput is high for a single node. **Verify that the RAID controller on the node can also meet or exceed the maximum throughput of the underlying storage media. If it cannot, size the network according to the maximum achievable throughput of the RAID controller.**

**Write-heavy environments**

Read and write operations produce different traffic patterns in a VxFlex OS environment. When a host (SDC) makes a single 4k read request, it must contact a single SDS to retrieve the data. The 4k block is transmitted once, out of a single SDS. If that host makes a single 4k write request, the 4k block must be transmitted to the primary SDS, then out of the primary SDS, then into the secondary SDS.

Write operations therefore require three times more bandwidth to SDSs than read operations. However, a write operation involves two SDSs, rather than the one required for a read operation. The bandwidth requirement ratio of reads to writes is therefore 1:1.5.

Stated differently, per SDS, a write operation requires 1.5 times more network throughput than a read operation when compared to the throughput of the underlying storage.

Under ordinary circumstances, the storage bandwidth calculations described earlier are sufficient. **However, if some of the SDSs in the environment are expected to host a write-heavy workload, consider adding network capacity.**

**Hyper-converged environments**

When VxFlex OS is in a hyper-converged deployment, each physical node is running an SDS, an SDC on the hypervisor, and one or more VMs. Hyper-converged deployments optimize hardware investments, but they also introduce network sizing requirements.

The storage bandwidth calculations described earlier apply to hyper-converged environments, but front-end bandwidth to any virtual machines, hypervisor or OS traffic, and traffic from the SDC, must also be taken into account. Though sizing for the virtual machines is outside the scope of this technical report, it is a priority.

In hyper-converged environments, it is also a priority to logically separate storage traffic from other network traffic.
An example of a hyper-converged VMware environment using 4 10 gigabit network connections. VxFlex OS traffic on this host uses ports Eth0 and Eth1. Redundancy is provided with native VxFlex OS IP multipathing, rather than MLAG. Ports Eth2 and Eth3 use both MLAG and VLAN tagging, and provide access network access to the hypervisor and the other guests. Other configurations are possible as VxFlex OS also supports VLAN tagging and link aggregation.

**Network Hardware**

**Dedicated NICs**

VxFlex OS engineering recommends the use of dedicated network adapters for VxFlex OS traffic, if possible. Dedicated network adapters provide dedicated bandwidth and simplified troubleshooting. Note that shared network adapters are fully supported, and may be mandatory in hyper-converged environments.

**Two NICs vs. Four NICs and Other Configurations**

VxFlex OS allows for the scaling of network resources through the addition of additional network interfaces. Although not required, there may be situations where isolating front-end and back-end traffic for the storage network may be ideal. This is a requirement in two-layer deployments where the storage and virtualization teams each manage their own networks. Another reason to segment front-end and back-end network traffic is to guarantee the performance of storage- and application-related network traffic. In all cases Dell EMC recommends multiple interfaces for redundancy, capacity, and speed.

PCI NIC redundancy is also a consideration. The use of two dual-port PCI NICs on each server is preferable to the use of a single quad-port PCI NIC, as a two dual-port PCI NICs can be configured to survive the failure of a single NIC.

**Switch Redundancy**

In most leaf-spine configurations, spine switches and top-of-rack (ToR) leaf switches are redundant. This provides continued access to components inside the rack in the network in the event a ToR switch fails. In cases where each rack contains a single ToR switch, ToR switch failure will result in an inability to access the SDS components inside the rack. Therefore, if a single ToR switch is used per rack, consider defining fault sets at the rack level to ensure data availability in the case of switch failure.

**Buffer Capacity**

To maximize VxFlex OS stability and performance the leaf switches should have a deep enough buffer size to protect against packet loss while a network is recovering from link or device failure.
**IP Considerations**

**IP-level Redundancy**

MDMs, SDSs, and SDCs can have multiple IP addresses, and can therefore reside in more than one network. This provides options for load balancing and redundancy.

VxFlex OS natively provides redundancy and load balancing across physical network links when an MDM or SDS is configured to send traffic across multiple links. In this configuration, each physical network port available to the MDM or SDS is assigned its own IP address, each in a different subnet.

The use of multiple subnets provides redundancy at the network level. The use of multiple subnets also ensures that as traffic is sent from one component to another, a different entry in the source component’s route table is chosen depending on the destination IP address. This prevents a single physical network port at the source from being a bottleneck as the source contacts multiple IP addresses (each corresponding to a physical network port) on a single destination.

Stated differently, a bottleneck at the source port may happen if multiple physical ports on the source and destination are in the same subnet. For example, if two SDSs share a single subnet, each SDS has two physical ports, and each physical port has its own IP address in that subnet, the IP stack will cause the source SDS to always choose the same physical source port. **Splitting ports across subnets allows for load balancing, because each port corresponds to a different subnet in the host’s routing table.**

A comparison of operating system IP configurations. The improper IP configuration on the left uses the same subnet, 10.10.10.0/24, for all traffic. When Server A initiates a connection to Server B, the network link providing a route to 10.10.10.0/24 will always be chosen for the outgoing connection. The second network port on Server A will not be utilized for outgoing connections. The proper IP configuration on the right uses two subnets, 10.10.10.0/24 and 192.168.1.0/24, allowing both ports on Server C to be utilized for outgoing connections. Note: the subnets chosen in this example (10.10.10.0/24 and 192.168.1.0/24) are arbitrary: the mixed use of a class “A” and a class “C” is meant for visual distinction only.

When each MDM or SDS has multiple IP addresses, VxFlex OS will handle load balancing more effectively due to its awareness of the traffic pattern. This can result in a small performance boost. Additionally, link aggregation maintains its own set of timers for link-level failover. Native VxFlex OS IP-level redundancy can therefore ease troubleshooting when a link goes down.

IP-level redundancy also protects against IP address conflicts. To protect against unwanted IP changes or conflicts, DHCP should not be deployed on a network where VxFlex OS MDMs or SDCs reside.

**Ethernet Considerations**

**Jumbo Frames**

While VxFlex OS supports jumbo frames, enabling jumbo frames can be challenging depending on your network infrastructure. Inconsistent implementation of jumbo frames by the various network components can lead to performance problems that are difficult to troubleshoot. When jumbo frames are in use, they must be enabled on every network component used by VxFlex OS infrastructure, including the hosts and switches.
Enabling jumbo frames allows more data to be passed in a single Ethernet frame. This decreases the total number of Ethernet frames and the number of interrupts that must be processed by each node. If jumbo frames are enabled on every component in your VxFlex OS infrastructure, there may be a performance benefit of approximately 10%, depending on your workload.

Because of the relatively small performance gains and potential for performance problems, Dell EMC recommends leaving jumbo frames disabled initially. **Enable jumbo frames only after you have a stable working setup and confirmed that your infrastructure can support their use.** Take care to ensure that jumbo frames are configured on all nodes along each path. Utilities like the Linux `tracepath` command can be used to discover MTU sizes along a path. Ping can be useful in diagnosing Jumbo Frame issues as well. On Linux, use the command of the form: `ping -M do -s 8972 <ip address/hostname>`.

(Note that here we are subtracting 28 bytes for un-encapsulated packet headers from the 9000 MTU size.)

### VLAN Tagging

VxFlex OS supports native VLANs and VLAN tagging on the connection between the server and the access or leaf switch. When measured by VxFlex OS engineering, both options provided the same level of performance.

### Link Aggregation Groups

Link Aggregation Groups (LAGs) and Multi-Chassis Link Aggregation Groups (MLAGs) combine ports between end points. The end points can be a switch and a host with LAG or two switches and a host with MLAG. Link aggregation terminology and implementation varies by switch vendor. MLAG functionality on Cisco Nexus switches is called Virtual Port Channels (vPC).

LAGs use the Link Aggregation Control Protocol (LACP) for setup, tear down, and error handling. LACP is a standard, but there are many proprietary variants.

Regardless of the switch vendor or the operating system hosting VxFlex OS, **LACP is recommended when link aggregation groups are used. The use of static link aggregation is not recommended.**

Link aggregation can be used as an alternative to IP-level redundancy, where each physical port has its own IP address. Link aggregation can be simpler to configure, and useful in situations where IP address exhaustion is an issue. Link aggregation must be configured on both the node running VxFlex OS and the network equipment it is attached to.

VxFlex OS is resilient and high performance regardless of the choice of IP-level redundancy or link aggregation. Performance of SDSs when MLAG is in use is close to the performance of IP-level redundancy.

- The choice of MLAG or IP-level redundancy for SDSs should be considered an operational decision.
- With MDM to MDM traffic, IP-level redundancy or LAG is strongly recommended over MLAG, as the continued availability of one IP address on the MDM helps prevent failovers, due to the short timeouts between MDMs, which are designed to communicate between multiple IP addresses.
- The use of LAG or MLAG is recommended for network ports hosting SDCs. This prevents a lengthy pause in I/O when a path is lost.

### LACP

LACP sends a message across each physical network link in the aggregated group of network links on a periodic basis. This message is part of the logic that determines if each physical link is still active. The frequency of these messages can be controlled by the network administrator using LACP timers.

LACP timers can typically be configured to detect link failures at a fast rate (one message per second) or a normal rate (one message every 30 seconds). When an LACP timer is configured to operate at a fast rate, corrective action is taken quickly. Additionally, the relative overhead of sending a message every second is small with modern network technology.

LACP timers should be configured to operate at a fast rate when link aggregation is used between a VxFlex OS SDS and a switch.

To establish an LACP connection, one or both of the LACP peers must be configured to use active mode. It is therefore recommended that the switch connected to the VxFlex OS node be configured to use active mode across the link.
Load Balancing

When multiple network links are active in a link aggregation group, the endpoints must choose how to distribute traffic between the links. Network administrators control this behavior by configuring a load balancing method on the end points. Load balancing methods typically choose which network link to use based on some combination of the source or destination IP address, MAC address, or TCP/UDP port.

This load-balancing method is referred to as a “hash mode”. Hash mode load balancing aims to keep traffic to and from a certain pair of source and destination addresses or transport ports on the same physical link, provided that link remains active.

The recommended configuration of hash mode load balancing depends on the operating system in use.

If a node running an SDS has aggregated links to the switch and is running Windows, the hash mode should be configured to use “Transport Ports”. This mechanism uses the source and destination TCP/UDP ports and IP addresses to load balance between physical network links.

If a node running an SDS has aggregated links to the switch and is running VMware ESX®, the hash mode should be configured to use “Source and destination IP address” or “Source and destination IP address and TCP/UDP port”.

If a node running an SDS has aggregated links to the switch and is running Linux, the hash mode on Linux should be configured to use the "xmit_hash_policy=layer2+3" or "xmit_hash_policy=layer3+4" bonding option. The "xmit_hash_policy=layer2+3" bonding option uses the source and destination MAC and IP addresses for load balancing. The "xmit_hash_policy=layer3+4" bonding option uses the source and destination IP addresses and TCP/UDP ports for load balancing.

On Linux, the “miimon=100” bonding option should also be used. This option directs Linux to verify the status of each physical link every 100 milliseconds.

Note that the name of each bonding option may vary depending on the Linux distribution, but the recommendations remain the same.

Multiple Chassis Link Aggregation Groups

Like link aggregation groups (LAGs), MLAGs provide network link redundancy. Unlike LAGs, MLAGs allow a single end point (such as a node running VxFlex OS) to be connected to multiple switches. Switch vendors use different names when referring to MLAG, and MLAG implementations are typically proprietary.

The use of MLAG is supported by VxFlex OS, but is not recommended for MDM to MDM traffic. The options described in the “Load Balancing” section also apply to the use of MLAG.

The MDM Network

Although MDMs do not reside in the data path between hosts (SDCs) and their distributed storage (SDSs), they are responsible for maintaining relationships between themselves to keep track of the state of the cluster. MDM to MDM traffic is therefore sensitive to network events that impact latency, such as the loss of a physical network link in an MLAG.

It is recommended that MDMs use IP-level redundancy on two or more network segments rather than MLAG. The MDMs may share one or more dedicated MDM cluster networks.

MDMs are redundant. VxFlex OS can therefore survive not just an increase in latency, but loss of MDMs. The use of MLAG to a node hosting an MDM will work. However, if you require the use of MLAG on a network that carries MDM to MDM traffic, please work with a Dell EMC VxFlex OS representative to ensure you have chosen a robust design.
Two nodes connected to two leaf switches. MDM traffic should traverse the purple links, because they are not in an MLAG group.

Network Services

DNS

The MDM cluster maintains the database of system components and their IP addresses. In order to eliminate the possibility of a DNS outage impacting a VxFlex OS deployment, the MDM cluster does not track system components by hostname or FQDN. If a hostname or FQDN is used when registering a system component with the MDM cluster, it is resolved to an IP address and the component is registered with its IP address.

Therefore, hostname and FQDN changes do not influence inter-component traffic in a VxFlex OS deployment.

Dynamic Routing Considerations

In large leaf-spine environments consisting of hundreds or thousands of nodes, the network infrastructure may be required to dynamically route VxFlex OS traffic.

A central objective to routing VxFlex OS traffic is to reduce the convergence time of the routing protocol. When a component or link fails, the router or switch must detect the failure; the routing protocol must propagate the changes to the other routers; then each router or switch must re-calculate the route to each destination node. If the network is configured correctly, this process can happen in less than 300 milliseconds: fast enough to maintain MDM cluster stability.

If the convergence time exceeds 400 milliseconds, the MDM cluster may fail over to a secondary MDM. The system will continue to operate and I/O will continue if the MDM fails over, but **300 milliseconds is a good target to maintain maximum system stability.** Timeout values for other system component communication mechanisms are much higher, so the system should be designed for the most demanding timeout requirements: those of the MDMs.

For the fastest possible convergence time, standard best practices apply. This means conforming to all network vendor best practices designed to achieve that end, including the absence of underpowered routers (weak links) that prevent rapid convergence.

Convergence time is insufficient in every tested network vendor’s default OSPF or BGP configuration. **Every routing protocol deployment, irrespective of network vendor, must include performance tweaks to minimize convergence time.** These tweaks include the use of Bidirectional Forwarding Detection (BFD) and the adjustment of failure-related timing mechanisms.

OSPF and BGP have both been tested with VxFlex OS. VxFlex OS is known to function without errors during link and device failures when routing protocols and networking devices are configured properly. However, **OSPF is recommended over BGP.** This recommendation is supported by test results that indicate OSPF converges faster than BGP when both are configured optimally for fast convergence.
Bidirectional Forwarding Detection (BFD)

Regardless of the choice of routing protocol (OSPF or BGP), the use of Bidirectional Forwarding Detection (BFD) is required. BFD reduces the overhead associated with protocol-native hello timers, allowing link failures to be detected quickly. BFD provides faster failure detection than native protocol hello timers for a number of reasons including reduction in router CPU and bandwidth utilization. **BFD is therefore strongly recommended over aggressive protocol hello timers.**

VxFlex OS is stable during network fail-overs when it is deployed with BFD and optimized OSPF and BGP routing. Sub-second failure detection must be enabled with BFD.

For a network to converge, the event must be detected, propagated to other routers, processed by the routers, and the routing information base (RIB) or Forwarding Information Base (FIB) must be updated. All these steps must be performed for the routing protocol to converge, and they should all complete in less than 300 milliseconds.

In tests using Cisco 9000 and 3000 series switches a **BFD hold down timer of 150 milliseconds** was sufficient. The configuration for a 150 millisecond hold down timer consisted of 50 millisecond transmission intervals, with a 50 millisecond min_rx and a multiplier of 3. The VxFlex OS recommendation is to use a maximum hold down timer of 150 milliseconds. If your switch vendor supports BFD hold down timers of less than 150 milliseconds, the shortest achievable hold down timer is preferred. BFD should be enabled in asynchronous mode when possible.

In environments using Cisco vPC (MLAG), **BFD should also be enabled on all routed interfaces and all host-facing interfaces running Virtual Router Redundancy Protocol (VRRP).**

An example of a BFD configuration on a Cisco Nexus switch. BFD is configured with a hold down timer of 150 milliseconds (the interval is 50 microseconds; the multiplier is 3). OSPF on interface port-channel51 and VRRP on interface Vlan30 are both configured as a client of BFD.

Physical Link Configuration

Timers involved with link failures are candidates for tuning. Link down and interface down event detection and handling varies by network vendor and product line. **On Cisco Nexus switches, “carrier-delay” timer should be set to 0 milliseconds on each SVI interface, and “link debounce” timer should both be set to 0 milliseconds on each physical interface.**

Carrier delay (carrier-delay) is a timer on the switch. It is applicable to an SVI interface. Carrier delay represents the amount of time the switch should wait before it notifies the application when a link failure is detected. Carrier delay is used to prevent flapping event notification in unstable networks. In modern leaf-spine environments, all links should be configured as point-to-point, providing a stable network. The recommended value for an SVI interface carrying VxFlex OS traffic is 0 milliseconds.

Debounce (link debounce) is a timer that delays link-down notification in firmware. It is applicable to a physical interface. Debounce is similar to carrier delay, but it is applicable to physical interfaces, rather than logical interfaces, and is used for link down notifications only. Traffic is stopped during the wait period. A nonzero link debounce setting can affect the convergence of routing protocols. The recommended value for a link debounce timer is 0 milliseconds for a physical interface carrying VxFlex OS traffic.
ECMP

The use of Equal-Cost Multi-Path Routing (ECMP) is required. ECMP distributes traffic evenly between leaf and spine switches, and provides high availability using redundant leaf to spine network links. ECMP is analogous to MLAG, but operates over layer 3 (IP), rather than over Ethernet.

ECMP is on by default with OSPF on Cisco Nexus switches. It is not on by default with BGP on Cisco Nexus switches, so it must be enabled manually. The ECMP hash algorithm used should be layer 3 (IP) or layer 3 and layer 4 (IP and TCP/UDP port).

OSPF

OSPF is the preferred routing protocol because when it is configured properly, it converges rapidly. When OSPF is used, the leaf and spine switches all reside in a single OSPF area. To provide stable, sub-300 millisecond convergence time, it is necessary to tune Link State Advertisement (LSA) and SPF timers. On all leaf and spine switches, the OSPF interfaces should be configured as point-to-point with the OSPF process configured as a client of BFD. All OSPF interfaces should be configured as point-to-point.

Link State Advertisements (LSAs)

Link State Advertisements (LSAs) are used by link state routing protocols such as OSPF to notify neighboring devices that the network topology has changed due to a device or link failure. LSA timers are configurable to prevent the network from being flooded with LSAs if a network port is flapping.

The LSA configuration on leaf and spine switches should be tuned with a start interval of 10 milliseconds or less. This means that if multiple LSAs are sourced from the same device, LSAs will not be sent more often than every 10 milliseconds.

The LSA hold interval on leaf and spine switches should be tuned to 100 milliseconds or less. This means that if a subsequent LSA needs to be generated within that time period, it will be generated only after the hold interval has elapsed. Once this occurs on a Cisco Nexus switch, the hold interval will then be doubled until it reaches the max interval. When the max interval is reached, the topology must remain stable for twice the max interval before the hold interval is reset to the start interval.

VxFlex OS testing was performed using a max interval of 5000 milliseconds (5 seconds). The max interval is less important than the start and hold interval settings, provided it is large enough to prevent excessive LSA traffic.

LSA arrival timers allow the system to drop duplicate copies of the same link state advertisement that arrive within the specified interval. The LSA arrival timer must be less than the hold interval. The recommended LSA arrival timer setting is 80 milliseconds.

Shortest Path First (SPF) Calculations

To prevent overutilization of router hardware during periods of network instability, shortest path first calculations can be delayed. This prevents the router from continually recalculating path trees as a result of rapid and continual topology fluctuations.

On Cisco Nexus switches, the algorithm that controls SPF timers is similar to the algorithm that controls LSAs. SPF timers should be throttled with a start time of 10 milliseconds or less and a hold time of 100 milliseconds or less. As with the LSA max interval, a max hold time of 5000 milliseconds was used under test, which is a reasonable default, but can be adjusted if needed.
OSPF configuration examples on a Cisco Nexus leaf switch (left) and spine switch (right). They are in the same OSPF area (100). All the interfaces running OSPF are configured point-to-point. BFD is configured on the OSPF router. SPF and LSA timers are configured to minimize convergence time in the event of link or switch failure.

**BGP**

Though OSPF is preferred because it can converge faster, BGP can also be configured to converge within the required time frame.

BGP is not configured to use ECMP on Cisco Nexus switches by default. It must be configured manually. If BGP is required, **EBGP is recommended over IBGP.** EBGP supports ECMP without a problem by default, but IBGP requires a BGP route reflector and the add-path feature to fully support ECMP.

EBGP can be configured in a way where each leaf and spine switch represents a different Autonomous System Number (ASN). In this configuration, each leaf has to peer with every other spine. Alternatively, EBGP can be configured such that all spine switches share same ASN and each leaf switch represents a different ASN.

**BGP leaf and spine switches should be configured for fast external failover (fast-external-failover on Cisco).** This command setting allows the switch to terminate BGP connections over a dead network link, without waiting for a hold down timer to expire.

Leaf and spine switches should also enable ECMP by allowing the switch to load share across multiple BGP paths, regardless of the ASN. On Cisco, this is done using the “best as-path multipath-relax” configuration. EBGP may require additional configuration parameters to enable ECMP. On Cisco, this includes setting the “maximum-path” parameter to number of available paths to spine switches.

**EBGP with VxFlex OS requires that BFD be configured on each leaf and spine neighbor.** When using BGP, the SDS and MDM networks are advertised by the leaf switch.
BGP configuration examples on a Cisco Nexus leaf switch (left) and spine switch (right). They reside in different autonomous systems (65123 and 65122). The "fast-external-failover" and "best as-path multipath-relax" options are enabled on both. The "maximum-path" parameter is tuned on both to match the number of paths to be used for ECMP (in this example, both are 4, but that may not always be the case). BFD is enabled for each leaf or spine neighbor. The leaf switch is configured to advertise the VxFlex OS MDM and SDS networks (20.20.20.0/24 and 30.30.30.0/24).

Host to Leaf Connectivity
Leaf switch failure is protected against by using multi-home topology with either a dual subnet configuration, or with MLAG.

Just as in environments where traffic is not dynamically routed, if MLAG is in use for SDS and SDC traffic, a separate IP network without MLAG is recommended for the MDM cluster. This increases stability by preventing MDM cluster failover when a link fails.
Leaf and Spine Connectivity

Configurations consisting of multiple uplinks between each leaf and spine switch are supported. In these configurations, each leaf switch is connected using multiple IP subnets to the same spine switch. Leaf switches can also be connected to a spine switch using bonded (aggregated) links. Aggregated links between leaf and spine switches use LAG, rather than MLAG. In a properly configured system, failover times for LAG are sufficient for MDM traffic.

Leaf to Spine Bandwidth Requirements

Assuming storage media is not a performance bottleneck, calculating the amount of bandwidth required between leaf and spine switches involves determining the amount of bandwidth available from each leaf switch to the attached hosts, discounting the amount if I/O that is likely to be local to the leaf switch, then dividing the remote bandwidth requirement between each of the spine switches.

Consider a situation with two racks where each rack contains two leaf switches and 20 servers, each server has two 10 gigabit interfaces, and each of these servers is dual-homed to the two leaf switches in the rack. In this case, the downstream bandwidth from each of the leaf switches is calculated as:

\[ 20 \text{ servers} \times 10 \text{ gigabits/\text{server}} = 200 \text{ gigabits} \]

The downstream bandwidth requirement for each leaf switch is 200 gigabits. However, some of the traffic will be local to the pair of leaf switches, and therefore will not need to traverse the spine switches.

The amount traffic that is local to the leaf switches in the rack is determined by the number of racks in the configuration. If there are two racks, 50% of the traffic will likely be local. If there are three racks, 33% of the traffic will likely be local. If there are four racks, 25% of the traffic is likely to be local, and so on. Stated differently, the proportion of I/O that is likely to be remote will be:

\[ \text{remote\_ratio} = \frac{\text{number\_of\_racks} - 1}{\text{number\_of\_racks}} \]

In this example, there are two racks, so 50% of the bandwidth is likely to be remote:

\[ \text{remote\_ratio} = \frac{2 \times \text{total\_racks} - 1 \text{ rack}}{2 \times \text{total\_racks}} = 50\% \]

Given that there are two racks in this example, 50% of the bandwidth is likely to be remote. Multiply the amount of traffic expected to be remote by the downstream bandwidth of each leaf switch to find the total remote bandwidth requirement from each leaf switch:

\[ \text{per\_leaf\_requirement} = 200 \text{ gigabits} \times 50\% \text{ remote\_ratio} = 100 \text{ gigabits} \]

100 gigabits of bandwidth is required between the leaf switches. However, this bandwidth will be distributed between spine switches, so an additional calculation is required. To find the upstream requirements to each spine switch from each leaf switch, divide the remote bandwidth requirement by the number of spine switches, since remote load is balanced between the spine switches.

\[ \text{per\_leaf\_to\_spine\_requirement} = \frac{\text{per\_leaf\_requirement}}{\text{number\_of\_spine\_switches}} \]

In this example, each leaf switch is expected to demand 100 gigabits of remote bandwidth through the mesh of spine switches. Since this load will be distributed among the spine switches, the total bandwidth between each leaf and spine is calculated as:

\[ \text{per\_leaf\_to\_spine\_requirement} = \frac{100 \text{ gigabits}}{2 \text{ spine\_switches}} = 50 \text{ gigabits/spine\_switch} \]

Therefore, for a nonblocking topology, two 40 gigabit connections for a total of 80 gigabits is sufficient bandwidth between each leaf and spine switch. Alternatively, five 10 gigabit connections from each leaf switch to each spine switch for a total of 50 gigabits is sufficient.
The equation to determine the amount of bandwidth needed from each leaf switch to each spine switch can be summarized as:

\[
\text{downstream\_bandwidth\_requirement} \times \left(\frac{(\text{number\_of\_racks} - 1)}{\text{number\_of\_spine\_switches}}\right)
\]

**VRRP Engine**

For routed access architectures with Cisco vPC and IP-level redundancy on the nodes, Dell EMC recommends using VRRP for the node default gateway. This allows the default gateway to fail over to the other leaf switch in the event of leaf switch failure.

**BFD should be enabled for each VRRP instance.** As with routing protocols, BFD allows VRRP to fail over quickly. It is recommended that VRRP be configured as primary on the active vPC peer and secondary on backup vPC Peer.

**AMS Considerations**

When AMS is used to configure new Dell EMC VxFlex OS Ready Nodes that reside in different subnets, an AMS discovery server is required on each subnet to act as a proxy. If no AMS proxy server exists on the subnet, AMS discovery will fail, and you will be unable to configure the new nodes through AMS.

**VMware Considerations**

Though network connections are virtualized in ESX, the same principles of physical network layout described in this document apply. Specifically, this means that MLAG should be avoided on links carrying MDM traffic unless a Dell EMC VxFlex OS representative has been consulted.

It is helpful to think of physical network from the perspective of the network stack on the virtual machine running the MDM or SDS, or the network stack in use by the SDC in the VMkernel. Considering the needs of the guest or host level network stack, then applying it to the physical network can inform decisions about the virtual switch layout.

**IP-level Redundancy**

When network link redundancy is provided using a dual subnet configuration, two separate virtual switches are needed. This is required because each virtual switch has its own physical uplink port. When VxFlex OS is run in hyper-converged mode, this configuration has 3 interfaces: VMkernel for the SDC, VM for the SDS, and uplink for physical network access. VxFlex OS natively supports installation in this mode.

**LAG and MLAG**

The use of the distributed virtual switch is required when LAG or MLAG is used. The standard virtual switch does not support LACP, and is therefore not recommended. When LAG or MLAG is used, the bonding is done on physical uplink ports.

VxFlex OS installation using the vSphere plugin does not natively support LAG or MLAG installation. It must be configured manually or automated using other means.
If a node running an SDS or SDC has aggregated links to a switch, the hash mode on the physical uplink ports should be configured to use “Source and destination IP address” or “Source and destination IP address and TCP/UDP port”.

**SDC**

The SDC is a kernel driver for ESX that implements the VxFlex OS client. Since it runs in ESX, it uses one or more VMkernel ports for communication with the other VxFlex OS components. If redundancy is desired, LAG or MLAG is recommended over a dual-subnet design.

**SDS**

The SDS is deployed as a virtual appliance on ESX. It can use IP-level redundancy, LAG, or MLAG.

**MDM**

The MDM is deployed as a virtual appliance on ESX. The use of IP-level redundancy is strongly recommended over the use of MLAG. *A single MDM should therefore use two or more separate virtual switches.*

### VXLAN Considerations

**Introduction**

VXLAN is conceptually similar to compute or storage virtualization. Its purpose is to decouple physical network infrastructure from virtual network infrastructure. In a leaf-spine topology (also called a Clos fabric); VXLANs allow physical nodes or virtual machines spread across a routed IP network to behave as if they were in isolated VLANs in an ordinary switched network.

VXLAN accomplishes this by encapsulating Ethernet frames in UDP. This allows the physical network that interconnects the datacenter infrastructure to be abstracted away from the tenants. The physical network is the “underlay” network. The virtualized networks visible to the tenants are the “overlay” networks. A hardware or software-based Virtual Tunnel End Point (VTEP) isolates tenants and moves tenant traffic into and out of the overlay network.

VXLAN preserves a machine’s LAN membership regardless of that machine’s physical location in a larger, routed IP network. It therefore enables things like continued IP reachability when a virtual machine is moved across physical IP subnets. It enables tenants on different LAN segments to use overlapping IP ranges, because the IP addresses in use by the tenants are not used to route traffic in the physical network infrastructure. VXLAN also enables uplink load balancing using Equal Cost Multipathing in the underlay network.
VXLAN expands the number of LANs available to tenants in the environment. A network using VXLANs can have more than 16 million virtual LAN segments as opposed to the 4094 LAN segments provided when VLANs alone are in use. Like VLANs, VXLANs isolate broadcasts such as ARP requests, restricting them to only those machines that reside in the same VXLAN.

Scalability, multitenancy, and virtualization make VXLAN a good choice for private cloud environments which are also well suited to VxFlex OS for block storage.

Since large VxFlex OS deployments using VXLAN are likely to be deployed in a Clos fabric, recommendations in the “Dynamic Routing Considerations” section of this document should be followed for the Clos fabric (and other sections). Depending on your environment, VXLAN may also depend on a multicast routing protocol such as PIM.

This document makes some specific recommendations for Cisco Nexus and VMware NSX® environments. Dell "S" and "Z" series switches support hardware-based VXLAN, and should be considered when planning your deployment. This document is not a substitute for vendor-recommended best practices. Vendor best practices, particularly those that speed protocol recovery following network device or link loss should be followed.

VXLAN Tunnel End Points (VTEPs)

A VXLAN Tunnel End Point (VTEP) is a logical device that may be implemented in hardware, software, or a mix. It has one or more IP addresses. When an IP network (such as an ordinary Clos fabric) carries VXLAN-encapsulated traffic, that traffic is routed to a VTEP.

The VTEP eliminates traditional Ethernet LAN boundaries by encapsulating Ethernet frames in UDP. This allows the physical network that interconnects the datacenter infrastructure to be abstracted away from the tenants. The network being abstracted is the “underlay” network. The virtualized network visible to the tenants is the “overlay” network.

A VXLAN traffic flow illustration. A physical or virtual machine on the left generates an Ethernet frame and sends it to a physical or virtual switch. Based on the destination MAC address in the Ethernet frame, the VTEP in the switch encapsulates the Ethernet frame in VXLAN, which is then encapsulated in UDP, then IP, sending it to the appropriate destination VTEP. The destination VTEP strips the VXLAN, UDP, and IP header used for transport through the underlay network, and then sends the frame to the physical or virtual destination port.

The headers prepended to the Ethernet frame by the VTEP contain VXLAN-specific fields. These fields include a VXLAN Network Identifier (VNI). A VNI is analogous to a VLAN tag. Physical or virtual machines mapped to the same VNI behave as if they are on the same LAN. The VNI field in VXLAN is 24 bits long instead of the 8 bits used in a VLAN tag, which is why more than 16 million VXLANs are addressable.

A VTEP may assign a VNI to an incoming Ethernet frame based on different criteria, depending on the product in use and its configuration. The VNI may be assigned based on a VLAN tag in the frame, the physical switch port, or the source MAC address.

The VXLAN Control Plane

The IP encapsulation between VTEPs described above is referred to as the “VXLAN data plane”. This encapsulation is conceptually simple, and is standardized by RFC 7348.

Coordinating information such as MAC addresses between remote VTEPs is handled by the “VXLAN control plane”. VXLAN control planes are less standardized and more complex. The control plane is responsible for handling broadcast, unknown unicast, and multicast (BUM) traffic. BUM traffic includes some traffic types handled natively by a LAN, such as Ethernet ARP requests.

One design objective, particularly in large environments, is to handle BUM traffic in an efficient manner. Since BUM traffic is communicated between VTEPs, not VXLAN tenants, the underlay (physical) network is used for communication. The mechanisms used to distribute BUM traffic depend on the networking products in use, and the features enabled.
One method for transmitting BUM traffic between VTEPs is through unicast traffic. Unicast traffic is a conversation between two IP addresses. It is the most common type of IP traffic. When BUM traffic is transmitted using unicast, a VTEP servicing a specific VXLAN must communicate directly with every other VTEP servicing that VXLAN. Unicast is appropriate in some circumstances, but unicast can also be expensive in large environments, both in terms of VTEP CPU and network bandwidth utilization.

Another method for transmitting BUM traffic is IP multicast. Multicast traffic is less common than unicast, and fewer people are familiar with multicast. IP multicasting allows a machine on an IP network (the VTEP, in this case) to subscribe to a feed of information coming from a multicast IP address in the range 224.0.0.0 through 239.255.255.255.

Multicast routing requires additional configuration of the underlay network. With multicast, the underlay network must be able to build a distribution tree to deliver messages to the clients that subscribe to the multicast. Since the multicast distribution tree depends upon the routes in use by the underlay network, we must include both routing protocol convergence time and multicast protocol convergence time when considering potential network link loss impact to VxFlex OS.

**Hardware based VXLAN**

Some physical network switches such as the Cisco Nexus 9000 series provide a hardware-based VXLAN gateway. These switches natively translate VLAN tagged Ethernet frames (IEEE 802.1Q) or traffic from a physical port into VXLAN traffic for transmission in the IP Clos underlay network.

**Ingress Replication**

When ingress replication is in use, leaf switches are configured to communicate VTEP state information directly between one another. When BUM traffic is encountered by one of the leaf switches, the CPU in the switch duplicates the packet and sends it to all other VTEPs, awaits the response from the VTEP serving the corresponding MAC address, and then forwards the response to the appropriate machine on one of its local LAN segments.

Ingress replication is CPU and bandwidth intensive, but may be appropriate in some smaller VxFlex OS environments. Protocol Independent Multicast (PIM), discussed in greater detail below, reduces protocol overhead by sending a single packet per subscriber link. PIM is therefore recommended over ingress replication for scalability. VxFlex OS engineering has verified ingress replication with up to 4 VTEPs.

A pair of Cisco Nexus 3172 leaf switches running VXLAN with ingress replication. VLAN 10 is mapped to VNI 8010. Any physical network port placed in VLAN 10, or any frame tagged as VLAN 10 will be transported over VNI 8010. Each switch has a virtual network interface, “nve1”, configured to use the IP address of loopback0. BUM traffic for VNI 8010 is configured for transmission to the other leaf switch in the pair.

Ingress replication has some requirements and limitations. In environments with multiple subnets, such as Clos fabrics, the use of a routing protocol like OSPF or BGP is required.
With ingress replication, a VNI can be associated only with a single IP address (the address of the VTEP), and an IP address can be associated only with a single VNI. This is not the case with multicast, where a single multicast address can carry BUM traffic for multiple VNIs.

Due to above limitations, VxFlex OS engineering recommends the use of Protocol Independent Multicast (PIM) over the use of unicast to forward BUM traffic in hardware-based VXLAN environments.

**Multicast**

BUM traffic in hardware based VXLAN can be handled using a unicast mechanism or a multicast mechanism. **Protocol Independent Multicast (PIM) running in sparse mode (PIM-SM) is the preferred mechanism for handling BUM traffic in a Cisco Nexus environment.**

PIM is an IP-specific multicast routing protocol. PIM is “protocol independent” in the sense that it is agnostic with respect to the routing protocol (BGP or OSPF, for example) running in the Clos fabric. Sparse mode refers to the mechanism in use by PIM to discover which downstream routers are subscribing to a specific multicast feed. Sparse mode is more efficient than PIM’s alternate “flood and learn” mechanism, which floods the network with multicast traffic periodically and prunes the distribution tree based on responses from multicast subscribers.

PIM uses a rendezvous point to distribute multicast traffic. In a Clos network, a rendezvous point is in a spine switch (a central location in the network) used to distribute PIM traffic. Rendezvous points can be configured for redundancy across the spine, and may move in the event of link or switch failure, thus the need to account for PIM convergence time in addition to routing protocol convergence time when considering network resiliency.

When PIM is used to transmit BUM traffic such as an ARP request, a message is sent out to all multicast subscribers for a specific VXLAN network segment (VNI). This message requests the MAC address for a specific IP address. All recipients of this multicast message passively learn the VTEP mapping for the source MAC address. The VTEP serving the destination MAC address responds to the VTEP making the query with a unicast message.

Two leaf switches configured to use Protocol Independent Multicast routing in Sparse Mode (PIM-SM) to send and receive VXLAN BUM traffic. VLAN 10 is mapped to VNI 8010 on leaf switch 1. VLAN 20 is mapped to VNI 8020 on leaf switch 2. Each switch has a virtual network interface, “nve1”, configured to use the IP address of loopback0. BUM traffic for VNI 8010 is configured for transmission through multicast group 224.1.1.10. BUM traffic for VNI 8020 is configured for transmission through multicast group 224.1.1.20. All other leaf switches configured in this way will receive multicast notifications for the associated VNIs (8010 or 8020).
A spine switch configured to act as a rendezvous point. All other spine switches will be configured this way. The rendezvous point is configured to carry a range of multicast groups in the range 224.1.1.0/24 (a range which includes the groups 224.1.1.10 and 224.1.1.20). PIM is enabled on the loopback interface for routing and on the interfaces connected to the spine switches.

VxFlex OS engineering has verified that VxFlex OS works well in a properly configured hardware-based VXLAN IP network fabric running the OSPF and PIM routing protocols. In this configuration, throughput was measured and is stable under various block sizes and read/write mixes. No performance degradation was measured with hardware-based VXLAN encapsulation.

VxFlex OS engineering measured the same performance with vPC and dual-subnet network topologies with hardware-based VXLAN. With vPC, the traffic was distributed evenly between the nodes that make up the fabric.

Note that, if the MDM Cluster is deployed on a different rack that is connected through the IP underlay network, then VxFlex OS engineering recommends the use of a separate, native IP network, without VXLAN, for maximum MDM cluster stability when a spine switch fails over. This is because during spine failover, the underlay routing protocol must converge, then PIM must converge on top of the newly established logical underlay topology.

Simplified illustrations of hardware-based VXLAN BUM traffic with ingress replication versus multicast. The VTEPs on leaves A, B, and D are all members of the same VXLAN, and are represented in purple. Both examples show a host in this VXLAN, attached to Leaf A, sending an Ethernet ARP request to the hosts attached to Leaf B and Leaf D.

When ingress replication is in use, two packets originate from Leaf A—one bound for Leaf B, the other bound for Leaf D. When multicast is in use, one packet is sent from Leaf A, is duplicated by a multicast rendezvous point in the spine, and is forwarded once down each subscriber link. The inefficiency with unicast gets worse at scale.

Note that these diagrams are simplified. The traffic may take the other spine switch, and the familiar term “packet” is used instead of the more correct term “datagram”.

**VXLAN with vPC**

VXLAN over Cisco Virtual Port Channels (vPC – the Cisco implementation of MLAG) is supported in Cisco Nexus environments. As with IP traffic that is not tunneled through VXLAN, the use of vPC is not recommended for MDM to MDM traffic. This is because the convergence time associated with link loss in a vPC configuration exceeds the timeouts associated with MDM to MDM traffic, and will therefore cause an MDM failover.

Dell EMC VxFlex OS Networking Best Practices and Design Considerations
© 2018 Dell Inc. or its subsidiaries.
When vPC is in use, each switch in the pair maintains its own VTEP. Therefore, **when vPC is in use, multicast must also be used.** This is because each VTEP in a pair of switches running vPC maintains its own overlay MAC address table. Therefore, in order to communicate changes to the vPC peer, multicast is required.

**Each switch in a vPC peer must also be configured to use the vPC peer-gateway feature.** vPC peer-gateway allows each switch in a vPC pair to accept traffic bound for the MAC address of the partner. This allows each switch to forward traffic without it first traversing the peer link between the vPC pair.

**Backup routing must be configured in each vPC peer.** This is so that if the uplink is lost between one of the peers, ingress traffic sent to this switch can be sent across the peer link, then up to the spine switch using the surviving uplinks. This is done by configuring a backup SVI interface with a higher routing metric between the vPC peers.

**Use a different primary loopback IP address for each vPC peer and same secondary IP address on both peers.** The secondary IP address is used as an anycast address to access either of the VTEPs in the pair, while the unique primary IP address is used by the system to access each VTEP individually.

**Enable IP ARP sync to improve routing protocol convergence time in vPC topologies.** ARP sync in a vPC pair helps quickly restore disparities between ARP tables in a vPC pair following the restoration of a switch or link after a brief outage or network channel flap.

**Other Hardware based VXLAN Considerations**

When hardware-based VXLAN is in use, **the host MTU should be configured to 50 bytes less than the switch MTU in order to avoid packet fragmentation.**

**Software based VXLAN with VMware NSX**

When software-based VXLAN is in use, the VTEPs run on the compute host. They can run as either as a hypervisor-based kernel module, virtual machine, or a combination of the two. VxFlex OS engineering has tested and verified the use of VMware NSX with VxFlex OS.

NSX can operate in one of three modes: multicast, unicast, and hybrid.

**Multicast mode** uses multicast in the underlay network to service BUM traffic. Multicast mode requires the use of PIM in the underlay network, so underlay configuration for multicast would be similar to that described in the “Multicast” section of this document.

**Unicast mode** completely decouples the underlay network from the virtual network. Unicast mode uses a virtualized control plane and it uses awareness of the physical subnets in the underlay network. In unicast mode, a VTEP (called a UTEP) in each underlay VLAN receives BUM traffic from the source VTEP and forwards it to every other interested VTEP in the physical underlay VLAN.

**In hybrid mode,** a VTEP (called an MTEP) in each underlay VLAN receives BUM traffic from the source VTEP, and uses Layer-2 multicasting via Internet Group Management Protocol (IGMP) to deliver it to the VTEPs on the underlay VLAN. Hybrid mode requires configuration of IGMP on the leaf switches in the underlay network, but does not require that the underlay network support PIM.
Simplified illustrations of VMware NSX VXLAN BUM traffic in unicast mode, multicast mode, and hybrid mode. VTEPs serving the same tenant VXLAN are represented in purple. These three examples show a virtual machine in this VXLAN, attached through Leaf A, sending an Ethernet ARP request to the virtual machines attached to Leaf B and Leaf D.

When unicast is in use, two packets originate from Leaf A – one bound for Leaf B, the other bound for Leaf D. Once the packet arrives on a VTEP (designated as the UTEP) in each of the VLANs on the leaf switches, that UTEP forwards the BUM traffic to the other interested VTEPs.

When multicast is in use, one packet is sent from Leaf A, is duplicated by a multicast (PIM) rendezvous point in the spine, and is forwarded once down each subscriber link. All interested VTEPs under the attached leaf switches receive the traffic, without it being duplicated across links or between VTEPs.

Hybrid mode is similar to unicast mode, but it uses IGMP on a VTEP (designated the MTEP) to distribute the traffic to the interested VTEPs in each underlay VLAN. The use of multicast reduces the number of packets sent by the MTEP versus those sent by a UTEP.

Note that these diagrams are simplified. The traffic may take the other spine switch, the familiar term “packet” is used instead of the more correct term “datagram”, and the broadcast and unicast messages travel through the leaf switch (not shown).
Software Considerations with NSX

VxFlex OS engineering recommends a separate NSX controller cluster used for management of the production deployment, including VxFlex OS. This is so that if the deployment network managed by NSX is broken, management connectivity to restore it is not lost.

Due to encapsulation of the VXLAN header, the MTU of the VxFlex OS virtual machine and the SDC ESXi VMkernel interface should be configured 50 bytes less than the MTU of the distributed virtual switch in order to avoid fragmentation.

Hardware and Performance Considerations with VMware NSX

VxFlex OS engineering has verified that VxFlex OS works properly on a Clos underlay network fabric with NSX VXLAN. Throughput was measured and determined to be stable under various block sizes and read/write mixes.

VxFlex OS engineering observed performance degradations between 12% and 25%, depending on the workload, with NSX software-based VXLAN in use. However, the NICs in use for measurement did not support VXLAN offloading. The use of NIC that support VXLAN offloading should improve the performance of the NSX VXLAN overlay network.

To take full advantage of the Intel® Converged Network Adapter's capabilities, enable Receive Side Scaling (RSS) in the ESXi host to balance the CPU load across multiple cores. See the VMware performance paper "VXLAN Performance Evaluation on VMware vSphere® 5.1" for performance results with the X520 Intel® Converged Network Adapter and RSS enabled.

Since connectivity between the host and the access switch is layer 2, the only option available for dual-home connectivity (load balancing and redundancy between leaf switches) with NSX is vPC or MLAG. As described in "The MDM Network" section of this document, it is recommended that MDMs use IP-level redundancy on two or more network segments rather than MLAG. In these cases, we recommend the use of separate, native IP networks, without VXLAN, for the MDM cluster for maximum MDM stability during leaf switch MLAG fail-over.

In Clos fabrics, even though underlay network convergence is not delivered by NSX, it still must be considered. However, when using NSX in unicast or hybrid mode, there is no multicast convergence to consider, just routing protocol (OSPF, BGP) convergence.

When vPC or LACP is enabled on the distributed virtual switch and the connected leaf switch, the recommendations for MLAG described in this document should be followed.

Validation Methods

VxFlex OS Native Tools

There are two main built-in tools that monitor network performance:

- SDS Network Test
- SDS Network Latency Meter Test

SDS Network Test

The SDS network test, "start_sds_network_test", is covered in the VxFlex OS User Manual. To fetch the results after it is run, use the "query_sds_network_test_results" command.

It is important to note that the parallel_messages and network_test_size_gb - options should be set so that they are at least 2x larger than the maximum network bandwidth. For Example: 1x10 gigabit NIC = 1250 megabytes * 2 = 2500 megabytes, or 3 gigabits rounded up. In this case you should run the command "--network_test_size_gb 3" This will ensure that you are sending enough bandwidth out on the network to have a consistent test result. The parallel message size should be equal to the total number of cores in your system, with a maximum of 16.
Example Output:

```bash
scli --start_sds_network_test --sds_ip 10.248.0.23
--network_test_size_gb 8 --parallel_messages 8
Network testing successfully started.

scli --query_sds_network_test_results --sds_ip 10.248.0.23
SDS with IP 10.248.0.23 returned information on 7 SDSs
  SDS 6bfc235100000000 10.248.0.24 bandwidth 2.4 GB (2474 MB) per-second
  SDS 6bfc235200000001 10.248.0.25 bandwidth 3.5 GB (3592 MB) per-second
  SDS 6bfc235400000003 10.248.0.26 bandwidth 2.5 GB (2592 MB) per-second
  SDS 6bfc235500000004 10.248.0.28 bandwidth 3.0 GB (3045 MB) per-second
  SDS 6bfc235600000005 10.248.0.30 bandwidth 3.2 GB (3316 MB) per-second
  SDS 6bfc235700000006 10.248.0.27 bandwidth 3.0 GB (3056 MB) per-second
  SDS 6bfc235800000007 10.248.0.29 bandwidth 2.6 GB (2617 MB) per-second
```

In the example above, you can see the network performance from the SDS you are testing to every other SDS in the network. Ensure that the speed per second is close to the expected performance of your network configuration.

SDS Network Latency Meter Test

The "query_network_latency_meters" command can be used to show the average network latency between SDS components. Low latency between SDS components is crucial for good write performance. When running this test, look for outliers and latency higher than a few hundred microseconds when 10 gigabit or better network connectivity is used. Note: this should be run from each SDS.

Example Output:

```bash
scli --query_network_latency_meters --sds_ip 10.248.0.23
SDS with IP 10.248.0.23 returned information on 7 SDSs

SDS 10.248.0.24
  Average IO size: 8.0 KB (8192 Bytes)
  Average latency (micro seconds): 231

SDS 10.248.0.25
  Average IO size: 40.0 KB (40960 Bytes)
  Average latency (micro seconds): 368

SDS 10.248.0.26
  Average IO size: 38.0 KB (38912 Bytes)
  Average latency (micro seconds): 315

SDS 10.248.0.28
  Average IO size: 5.0 KB (5120 Bytes)
  Average latency (micro seconds): 250

SDS 10.248.0.30
  Average IO size: 1.0 KB (1024 Bytes)
  Average latency (micro seconds): 211

SDS 10.248.0.27
  Average IO size: 9.0 KB (9216 Bytes)
  Average latency (micro seconds): 252

SDS 10.248.0.29
  Average IO size: 66.0 KB (67584 Bytes)
  Average latency (micro seconds): 418
```
Iperf, NetPerf, and Tracepath

**NOTE**: Iperf and NetPerf should be used to validate your network before configuring VxFlex OS. If you identify issues with Iperf or NetPerf, there may be network issues that need to be investigated. If you do not see issues with Iperf/NetPerf, use the VxFlex OS internal validation tools for additional and more accurate validation.

Iperf is a traffic generation tool, which can be used to measure the maximum possible bandwidth on IP networks. The Iperf feature set allows for tuning of various parameters and reports on bandwidth, loss, and other measurements. When Iperf is used, it should be run with multiple parallel client threads. Eight threads per IP socket is a good choice.

NetPerf is a benchmark that can be used to measure the performance of many different types of networking. It provides tests for both unidirectional throughput, and end-to-end latency.

The Linux “tracepath” command can be used to discover MTU sizes along a path.

Network Monitoring

It is important to monitor the health of your network to identify any issues that are preventing your network for operating at optimal capacity, and to safeguard from network performance degradation. There are a number of network monitoring tools available for use on the market, which offer many different feature sets.

Dell EMC recommends monitoring the following areas:

- Input and output traffic
- Errors, discards, and overruns
- Physical port status

Network Troubleshooting Basics

- Verify connectivity end-to-end between SDSs and SDCs using ping
- Test connectivity between components in both directions
- SDS and MDM communication should not exceed 1 millisecond network-only round-trip time. Verify round-trip latency between components using ping
- Check for port errors, discards, and overruns on the switch side
- Verify VxFlex OS nodes are up
- Verify VxFlex OS processes are installed and running on all nodes
- Check MTU across all switches and servers, especially if using jumbo frames
- Prefer 10 gigabit Ethernet over 1 gigabit Ethernet when possible
- Check for NIC errors, high NIC overrun rates (more than 2%), and dropped packets in the OS event logs
- Check for IP addresses without a valid NIC association
- Verify the network ports needed by VxFlex OS are not blocked by the network or the node
- Check for packet loss on the OS running VxFlex OS using event logs or OS network commands
- Verify no other applications running on the node are attempting to use TCP ports required by VxFlex OS
• Set all NICs to full duplex, with auto negotiation on, and the maximum speed supported by your network
• Check VxFlex OS native tool test outputs
• Check for RAID controller misconfiguration (this is not network, but it is a common performance problem)
• If you have a problem, collect the logs as soon as you can before they are over-written
• Additional troubleshooting, log collection information, and an FAQ is available in the VxFlex OS User Guide

Summary of Recommendations

Traffic Types
• Familiarize yourself with VxFlex OS traffic patterns to make informed choices about the network layout.
• To achieve high performance, low latency is important for SDS to SDS and SDC to SDS traffic.
• Pay special attention to the networks that provide MDM to MDM connectivity, prioritizing low latency and stability.

Network Infrastructure
• Use a non-blocking network design.
• Use a leaf-spine network architecture without uplink oversubscription if you plan to scale beyond a small deployment.

Network Performance and Sizing
• Ensure SDS and MDM components in the system have 1 millisecond round-trip network response times between each other under normal operating conditions.
• Use 10 gigabit network technology at a minimum.
• Use redundant server network connections for availability.
• SDS components that make up a protection domain should reside on hardware with equivalent storage and network performance.
• Size SDS network throughput using media throughput as a guide.
• Convert megabytes (storage native throughput metric) to gigabits (network native throughput metric) as follows:
  \[
  \text{gigabits} = \frac{\text{megabytes} \times 8}{1,000}
  \]
• Size network throughput based on the best achievable performance of the underlying storage media. If the RAID controller will bottleneck storage throughput, size network performance based on the RAID controller.
• If the workload is expected to be write-heavy, consider adding network bandwidth.
• If the deployment will be hyper-converged, front-end bandwidth to any virtual machines, hypervisor or OS traffic, and traffic from the SDC must also be taken into account.

Network Hardware
• Use dedicated network adapters for VxFlex OS traffic, if possible.
• In two-layer deployments front-end (SDC, SDS, MDM) and back-end (SDS, MDM) traffic resides on separate networks. This pattern isolates rebuild and rebalance traffic from client traffic.
• In cases where each rack contains a single ToR switch, consider defining fault sets at the rack level.

• The use of two dual-port PCI NICs on each server is preferable to the use of a single quad-port PCI NIC, as a two dual-port PCI NICs can be configured to survive the failure of a single NIC.

**IP Considerations**

• Splitting ports across subnets allows for load balancing, because each port corresponds to a different subnet in the host’s routing table.

• IP-level redundancy is preferred, but not strongly preferred, over MLAG for links in use by SDS components. The choice of IP-level redundancy or MLAG should be considered an operational decision. LAG or MLAG is preferred for SDC components.

• IP-level redundancy is strongly preferred over MLAG for links in use for MDM to MDM communication.

**Ethernet Considerations**

• If the use of jumbo frames is desired, enable them only after you have a stable working setup and you have confirmed that your infrastructure can support them.

**Link Aggregation Groups**

• The use of LACP is recommended. The use of static link aggregation is not recommended.

• When link aggregation is used between a VxFlex OS SDS and a switch, use LACP fast mode.

• Configure the switch ports attached to the VxFlex OS node to use active mode across the link.

• If a node running an SDS has aggregated links to the switch and is running Windows, the hash mode should be configured to use “Transport Ports”.

• If a node running an SDS has aggregated links to the switch and is running VMware ESX®, the hash mode should be configured to use “Source and destination IP address” or “Source and destination IP address and TCP/UDP port”.

• If a node running an SDS has aggregated links to the switch and is running Linux, the hash mode on Linux should be configured to use the "xmit_hash_policy=layer2+3" or "xmit_hash_policy=layer3+4" bonding option. On Linux, the “miimon=100” bonding option should also be used.

**The MDM Network**

• Prefer IP-level redundancy on two or more network segments rather than MLAG.

• Work with a Dell EMC VxFlex OS representative if you wish to use MLAG on ports delivering MDM traffic.

**Network Services**

• Hostname and FQDN changes do not influence inter-component traffic in a VxFlex OS deployment, because components are registered with the system using IP addresses.

**Dynamic Routing Considerations**

• The routing protocol should converge within 300 milliseconds to maintain maximum MDM stability.

• Every routing protocol deployment must include performance tweaks to minimize convergence time.

• OSPF is preferred because it can converge faster than BGP. Both can converge within the required time.

• Use BFD over protocol native hello timers.

• Use short BFD hold down timers (150 milliseconds or less).

• BFD should be enabled on all routed interfaces and all host-facing interfaces running Virtual Router Redundancy Protocol (VRRP).
• Set carrier link and debounce timers to 0 on all routed interfaces.
• Use ECMP with layer 3 or layer 3 and layer 4 hash algorithms for load balancing.
• Configure all OSPF interfaces as point-to-point.
• With OSPF, configure LSA timers appropriately, with a start, hold, and wait time of 10, 100, and 5000 milliseconds, respectively.
• With OSPF, configure SPF timers appropriately, with a start, hold, and wait time of 10, 100, and 5000 milliseconds, respectively.
• Verify that ECMP is configured for EBGP.
• Prefer EBGP over IBGP, as IBGP requires additional configuration for ECMP.
• With BGP, verify that leaf and spine switches are configured for fast external failover.
• With BGP, verify that the SDS and MDM networks are advertised by the leaf switch.
• For a non-blocking topology, size the connections from leaf to spine switches appropriately. The equation to determine the amount of bandwidth needed from each leaf switch to each spine switch can be summarized as:

\[
\frac{\text{downstream_bandwidth\_requirement} \times ((\text{number\_of\_racks} - 1) / \text{number\_of\_racks})}{\text{number\_of\_spine\_switches}}
\]

• When using AMS, a discovery server is required on each subnet to act as a proxy.

Hardware-based VXLAN Considerations
• Follow the recommendations in this document for the underlay Clos fabric.
• Follow vendor best practices, particularly those that speed protocol recovery following network device or link loss.
• Configure the system to handle BUM traffic in an efficient manner.
• Include both routing protocol convergence time and multicast protocol convergence time when considering potential network link loss impact to VxFlex OS.
• PIM-SM is the preferred mechanism for handling BUM traffic in a Cisco Nexus environment.
• If the MDM Cluster is deployed on a different rack that is connected through the IP underlay network, use a separate, native IP network, without VXLAN, for maximum MDM cluster stability.
• When vPC is in use, multicast must also be used.
• Each switch in a vPC peer must be configured to use the vPC peer-gateway feature.
• Backup routing must be configured in each vPC peer.
• Use a different primary loopback IP address for each vPC peer and same secondary IP address on both peers.
• Enable IP ARP sync to improve routing protocol convergence time in vPC topologies.
• The host MTU should be configured to 50 bytes less than the switch MTU in order to avoid packet fragmentation.

Software-based VXLAN Considerations
• Use a separate NSX controller cluster for the management plane of the production deployment.
• The MTU of the VxFlex OS virtual machine and the SDC ESXi VMkernel interface should be 50 bytes less than the MTU of the distributed virtual switch.
• NICs that support VXLAN offloading should improve the performance of the NSX VXLAN overlay network.
• Enable Receive Side Scaling (RSS) in the ESXi host to balance the CPU load across multiple cores.
• Use IP-level redundancy on two or more network segments rather than MLAG (or vPC) for the MDM.
• Even though underlay network convergence is not delivered by NSX, it still must be considered for timing purposes.
• When vPC or LACP is enabled on the distributed virtual switch and the connected leaf switch, the recommendations for MLAG described in this document should be followed.

**VMware Considerations**

• When network link redundancy is provided using a dual subnet configuration, two separate virtual switches are needed.
• The use of the distributed virtual switch is required when LAG or MLAG is used. Link aggregation must be enabled on the distributed switch.
• When using IP-level redundancy, a single MDM should use two or more separate virtual switches. These switches may be shared with SDS components on a converged network.

**Validation Methods**

• VxFlex OS provides the native tools "scli --start_sds_network_test" and "scli --query_network_latency_meters".
• The Iperf and NetPerf tools can be used to measure network performance.
• Follow the recommendations listed in the document for troubleshooting network-related VxFlex OS issues.

**Conclusion**

The selected deployment option, network topology, performance requirements, Ethernet, dynamic IP routing, and validation methods, all factor into a robust and sustainable network design. Dell EMC VxFlex OS can scale to 1024 nodes containing a variety of storage media, so one should size the network installation for future growth. The fact that VxFlex OS can be deployed in a hyper-converged mode where compute and storage reside on the same set of nodes, or in a two-layer mode where storage and compute resources are separate affect your decisions as well. To achieve immense performance, scalability, and flexibility, the network must be designed to account for the needs of the business. Following the principles and recommendations in this guide will result in a resilient, massively scalable, and high-performing block storage infrastructure.