

NVIDIA DGX BasePOD

Deployment Guide Featuring NVIDIA DGX A100 Systems

Document History

Version	Date	Authors	Description of Change
01	2022-11-12	Davinder Singh, Greg Zynda, Robert Sohigian, Robert Strober, Scott Ellis, and Yang Yang	Initial release
02	2022-11-17	Robert Sohigian and Yang Yang	Procedural updates
03	2023-03-15	Robert Sohigian and Yang Yang	Procedural and formatting updates
04	2023-04-07	Joe Handzik, Michael Levy, and Robert Sohigian	Further procedural and formatting updates

DG-11192-001

Abstract

Artificial intelligence (AI) infrastructure requires significant compute resources to train the latest state-of-the-art models efficiently, often requiring multiple nodes running in a distributed cluster.

While cloud computing provides an easy on-ramp to train AI models, many enterprises require an on-premises data center for a variety of technical or business reasons.

Building Al infrastructure on-premises can be a complex and confusing process. Careful planning and coordination will make the cluster deployment easier, and the job of the cluster administrators tasked with the day-to-day operation easier.

NVIDIA DGX BasePOD[™] provides the underlying infrastructure and software to accelerate deployment and execution of these new AI workloads. By building upon the success of NVIDIA DGX[™] systems, DGX BasePOD is a prescriptive AI infrastructure for enterprises, eliminating the design challenges, lengthy deployment cycle, and management complexity traditionally associated with scaling AI infrastructure.

The DGX BasePOD is built upon the <u>NVIDIA DGX A100 system</u>, which offers unprecedented compute performance with eight NVIDIA A100 Tensor Core GPUs connected with NVIDIA NVLink[®] and NVIDIA NVSwitch[™] technologies for fast inter-GPU communication.

Powered by NVIDIA Base Command[™], DGX BasePOD provides the essential foundation for AI development optimized for the enterprise.

Contents

Chapter 1.	Architecture	1
1.1 Har	dware Overview	1
1.2 Net 1.2.1 1.2.2 1.2.3 1.2.4 1.2.5	working DGX A100 System Network Ports DGX BasePOD Network Overview internalnet and externalnet ipminet ibnet	2 2 4 5 5
1.3 Sof 1.3.1 1.3.2	tware K8s Jupyter (Optional)	6 6 6
1.4 Sto	rage	7
Chapter 2.	Deployment	8
2.1 Clus 2.1.1 2.1.2	ster Configuration Network Configuration Configure Disk Layouts for Node Categories	
2.1.2.	2 Configure Disk Layout for the DGX Node Category	
2.1.3	Configure Node Network Interfaces	
2.1.3.	Configure Bright to Allow MAC Addresses to PXE Boot	
2.1.3.2	2 Configure Provisioning Interfaces on the DGX Nodes	
2.1.3.4	4 Configure InfiniBand Interfaces on DGX Nodes	
2.1.3.5	5 Identify the Cluster Nodes	
2.1.3.6	6 Update the Software Images	
2.2 Pow	ver On and Provision Cluster Nodes	35
2.3 Dep	loy Docker	
2.4 Dep 2.4.1 2.4.1. 2.4.1.2 2.4.1.3	loy K8s Install the Network Operator Preparation Deploy the Network Operator Run a Multi-node NCCL Test	
2.5 (Op 2.5.1	tional) Deploy Bright Jupyter Install Jupyter Using the CLI Wizard	63 63
Chapter 3. 3.1.1	High Availability Verify HA Setup	67 84

Chapter	⁻ 4.	Basic User Management	87
4.1 4.1.	Con 1	figuring a User Procedures to Remove a User	87 88
4.2	Add	ing a User to K8s	89
4.3	Rem	noving a User from K8s	90
Append	ix A.	Site Survey	v
Append	ix B.	Switch Configurations	x
B.1	SN4	600 #1 (In-band Management Switch)	x
B.2	SN4	600 #2 (In-band Management Switch)	xiii
B.3	SN2	201 (Out-of-band Management Switch)	xvi
B.4	Ethe	ernet Network Configuration Verifications	xvi

Chapter 1. Architecture

1.1 Hardware Overview

The DGX BasePOD consists of DGX A100 compute nodes, five control plane servers (two for cluster management and three Kubernetes (K8s) master nodes), as well as associated storage and networking infrastructure.

An overview of the hardware is in Table 1. Details about the hardware that can be used and how it should be cabled are given in the <u>NVIDIA DGX BasePOD Reference</u> <u>Architecture</u>.

This deployment guide describes the steps necessary for configuring and testing a fournode DGX BasePOD after the physical installation has taken place.

Component	Technology
Compute nodes	DGX A100 system
Compute fabric	NVIDIA Quantum QM8700 HDR 200 Gbps InfiniBand
Management fabric	NVIDIA SN4600 switches
Storage fabric	NVIDIA SN4600 switch for Ethernet attached storage NVIDIA Quantum QM8700 HDR 200 Gb/s for InfiniBand attached storage
Out-of-band management fabric	NVIDIA SN2201 switches
Control plane	 Minimum Requirements (each server): 64-bit x86 processor, AMD EPYC 7272 or equivalent 256 GB memory 1 TB SSD Two 100 Gbps network ports

1.2 Networking

This section covers the DGX system network ports and an overview of the networks used by DGX BasePOD.

1.2.1 DGX A100 System Network Ports

Figure 1 shows the rear of the DGX A100 system with the network port configuration used in this solution guide.

The following ports are selected for DGX BasePOD networking:

- Four single-port ConnectX-6 cards are used for the InfiniBand compute fabric, two on either side of the chassis (marked in red).
- > Two ports of the dual-port ConnectX-6 cards are configured as a bonded Ethernet interface for in-band management and storage networks. These are the bottom port from slot 4 and the right port from slot 5 (marked in blue).
- > BMC network access is provided through the out-of-band network (marked in gray).



Figure 1. DGX A100 system rear

Table 2 details the naming and mapping for the ConnectX-6 interfaces.

Slot Number	PCI Bus Number	Port Destination	InfiniBand RDMA Port
0	0000:4b:00.0	ibp75s0	mlx5_1
2	0000:ba:00.0	ibp186s0	mlx5_5
4 top	0000:e1:00.0	ibp225s0f0	mlx5_6
4 bottom	0000:e1:00.1	enp225s0f1	mlx5_7
5 left	0000:61:00.0	ibp97s0f0	mlx5_2
5 right	0000:61:00.1	enp97s0f1	mlx5_3
6	0000:0c:00.0	ibp12s0	mlx5_0
8	0000:8d:00.0	ibp141s0	mlx5_4

Table 2. Interface naming and mapping

1.2.2 DGX BasePOD Network Overview

There are four networks in a DGX BasePOD configuration:

- internalnet—Network used exclusively within the cluster, for storage and in-band management.
- externalnet—Network connecting the DGX BasePOD to an external network, such as a corporate or campus network.
- > ipminet—Network for out of band management, connecting BMCs.
- ibnet—InfiniBand network connecting all DGX systems ConnectX-6 Compute Fabric HCAs.

These are shown in Figure 2.

Figure 2. Network design and topology diagram



1.2.3 internalnet and externalnet

internalnet uses VLAN 122 and externalnet uses VLAN 121. Both VLANs are configured on the SN 4600 switches, which are the backbone of the DGX BasePOD Ethernet networking. Each DGX system connects to the SN4600 switches with a bonded interface consist of two physical interfaces, slot 4 bottom port (storage 4-2) and slot 5 right port (storage 5-2).

The K8s master nodes and the NFS storage device have a similar bonded interface configuration connected to SN4600 switches. Two SN4600 switches with Multi-chassis Link Aggregation (MLAG) provides the redundancy for DGX systems, K8s master nodes, and other devices with bonded interfaces. Trunk mode is used to bond the interface with VLAN 122 as its native VLAN. Access mode is used on the port connected to the Bright head node. BGP protocols used between interfaces are described in Table 3. All connected subnets are redistributed into BGP.

Protocol	Description
BGP	Used as required for routing between switches
iBGP	Configured between the two SN4600s using the MLAG peerlink.4094 interface
eBGP	Configured between the uplink (SN4600) and IPMI (SN2201) switches

Table 3. BGP protocols

1.2.4 ipminet

On the ipminet switches, the gateway for VLAN 111 is configured and all the ports connected to the end hosts are configured as access ports for VLAN 111. Each Bright head node requires two interfaces connected to the IPMI switch; the first for IPMI interface of the host and the second to be used as HOST OS's direct access to the IPMI subnet. Uplinks are connected to TOR-01 and TOR-02 using unnumbered eBGP. All connected subnets are redistributed into BGP. IPMI switch can also be uplinked to separate management network if required, rather than the TOR switches; still IPMI subnet route must be advertised to the in-band network so that Bright can control hosts using the IPMI network.

1.2.5 ibnet

For the ibnet, NICs on physical DGX slot 0 and 2 are connected to QM8700-1 InfiniBand switch; and the NICs on physical DGX slot 6 and 8 are connected to QM8700-2 InfiniBand switch. To manage the InfiniBand fabric, a subnet manager is required; one of the 8700 switches must be configured as the subnet manager.

1.3 Software

Base Command Manager (BCM) is a key software component of DGX BasePOD. BCM is used to provision the OS on all hosts, deploy K8s, optionally deploy Jupyter, and provide monitoring and visibility of the cluster health.

An instance of BCM runs on a pair of head nodes in an HA configuration and is connected to all other nodes in the DGX BasePOD.

DGX systems within a DGX BasePOD have a DGX OS image installed by BCM. Similarly, the K8s master nodes are imaged by BCM with an Ubuntu LTS version equivalent to that of the DGX OS and the head nodes themselves.

1.3.1 K8s

K8s is an open-source platform for automating deployment, scaling, and operations of application containers across clusters of hosts. With K8s, it is possible to:

- > Scale applications on the fly.
- > Seamlessly update running services.
- > Optimize hardware availability by using only the needed resources.

The cluster manager provides the administrator with the required packages, allows K8s to be set up, and manages and monitors K8s.

1.3.2 Jupyter (Optional)

BCM can optionally deploy and manage Jupyter, consisting of four major components and several extensions. The major components are: Jupyter Notebook, JupyterLab, JupyterHub, and Jupyter Enterprise Gateway.

These are the Jupyter extensions that BCM deploys:

- Template specialization extension—create custom Jupyter kernel without editing text files.
- > Job management extension—manage jobs from within the Jupyter interface.
- > VNC extension—interact with the X display of the execution server (including the desktop) from within the Jupyter interface.
- > K8s operators—Jupyter kernel, PostgreSQL, and Spark operators.
- > Jupyter dev server—Proxy server that enables developing applications in alternative editors while the computational workload is proxied to their Jupyter notebook running on the cluster.

1.4 Storage

An NFS server is required for a highly available (HA) BCM installation, and the required export path for that is described in this DGX BasePOD document. A DGX BasePOD typically also includes dedicated storage, but the configuration of that is outside the scope of this document. Contact the vendor of the storage solution being used for instructions on configuring the high-performance storage portions of a DGX BasePOD.

Chapter 2. Deployment

Deployment of a DGX BasePOD involves pre-setup, deployment, and use of BCM to provision the K8s cluster, and optionally deploy Jupyter.

1. Prepare the infrastructure.

Physical installation should be completed before using this document, along with capturing information about the intended deployment in a site survey. Refer to Appendix A for the example site survey used by this document.

2. Configure the networking switches.

Refer to Appendix B for the example configuration used by this document. Specifics on connecting to and configuring the switches can be found in their associated user guides.

- 3. Configure the NFS server.
 - a. As stated in Section 1.4, NFS configuration steps are not in scope for this document.
 - b. This DGX BasePOD deployment uses the /var/nfs/general, which is the NFS export path provided in Table 3 of the Site Survey.

c. Use the following parameters for the NFS server export file /etc/exports /var/nfs/general *(rw,sync,no_root_squash,no_subtree_check)

4. Set the DGX BIOS so that the DGX systems PXE boot by default.

BCM requires DGX systems to PXE boot.

a. Connect to the BMC of the DGX system.

b. In the Network tab of the System Inventory window, locate the MAC addresses for the Storage 4-2 and Storage 5-2 interfaces.

NVIDIA DGX [™] A100	≡			🛕 🗘 Sync
Uptime : 48 hr, 27 min, 34 sec FW : 0.16.09 IP : MAC	System Inventory		X	A
Chassis Part . Chassis SN Host Online Chassis Identify LED	Memory Controller Storage Network NIC Info			
Quick Links	Name	MACAddress		BDF
🕈 Dashboard	Cluster 0	8:C0:EB:97:25:86		8D:00:0
🕫 Sensor	Cluster 1	8:C0:EB:97:25:E2		BA:00:0
	Cluster 6	8:C0:EB:97:25:1E		0C:00:0
 System Inventory 	Cluster 7	8:C0:EB:97:25:12		4B:00:0
FRU Information	Cluster 8	5C:FF:35:FB:74:B9		E2:00:0
GPU Information	Storage 4-1	C:42:A1:A:33:4A		E1:00:0
네 Logs & Reports >	Storage 4-2	C:42:A1:A:33:4B		E1:00:1
	Storage 5-1	C:42:A1:74:F3:1E		61:00:0
🗘 Settings	Storage 5-2	C:42:A1:74:F3:1F		61:00:1

c. In the DGX A100 system BIOS, configure Boot Option #1 to be [NETWORK]. Set other Boot devices to [DISABLED].

Aptio Setup Utility - Main Advanced Chipset	- Copyright (C) 2021 Americ Security Boot Save & Ex	c <mark>an Megatrends, Inc.</mark> kit Server Mgmt
Boot Configuration Setup Prompt Timeout Bootup NumLock State	1 [0n]	Sets the system boot order
FIXED BOOT ORDER Priorit. Boot Option #1 Boot Option #2 Boot Option #3 Boot Option #4 Boot Option #5 Boot Option #6 • UEFI NETWORK Drive BBS Pr	<pre>++: Select Screen t1: Select Item Enter: Select +/-: Change Opt. F1: General Help F2: Previous Values F3: Optimized Defaults F4: Save & Exit ESC: Exit</pre>	

d. Disable PXE boot devices except for <code>storage 4-2</code> and <code>storage 5-2</code>. Set them to use IPv4.

Aptio Setup Utility – Copyright (C) 2021 American Megatrends, Inc. Boot				
Root	Ontion	<i>#</i> 1		Sats the sustem boot
0000	option	u T	Mellanov Network	order
			Adapter -	or der
			0C:42:A1:0A:33:4B]	
Boot	Option	#2	[UEFI: PXE IPv4	
			Mellanox Network	
			Adapter –	
			0C:42:A1:74:F3:1F]	
Boot	Option	#3	[Disable]	
Boot	Option	#4	[Disable]	
Boot	Option	#5	[Disable]	↔+: Select Screen
Boot	Option	#6	[Disable]	↑↓: Select Item
Boot	Option	#7	[Disable]	Enter: Select
Boot	Option	#8	[Disable]	+/−: Change Opt.
Boot	Option	#9	[Disable]	F1: General Help
Boot	Option	#10	[Disable]	F2: Previous Values
Boot	Option	#11	[Disable]	F3: Optimized Defaults
Boot	Option	#12	[Disable]	F4: Save & Exit
				ESU: EXIT
	Version	2 20 1275	Copuright (C) 2021 American	Megatrends Inc
	VCI 5101	1 2.20.1213. 1	Sopariant (6) 2021 Milerica	AB

e. Configure a static IP address for the BMC.

Navigate to the Server Mgmt tab of the BIOS, enter the BMC network configuration menu, then set the IPv4 Lan channel 1 Configuration Address Source option to StaticAddress, enter the IP address, subnet, and gateway/router information.

Aptio Setup Utility	– Copyright (C) 2021 Ameri	ican Megatrends, Inc. Server Mgmt
BMC network configurat жжжжжжжжжжжжжжжжжж Configure IPv4 support жжжжжжжжжжжжжжжжжж Lan channel 1 Configuration Address source	(Unspecified)	Select to configure LAN channel parameters statically or dynamically(by BIOS or BMC). Unspecified option will not modify any BMC network parameters during BIOS
Address source Station IP address Subnet mask Station MAC address Router IP address Router MAC address Lan channel 2	10.130.111.70 255.255.255.192 5C-FF-35-E1-9A-C2 10.130.111.65 68-21-5F-4F-1A-01	<pre>++: Select Screen 11: Select Item Enter: Select +/-: Change Opt. F1: General Help F2: Previous Values F3: Optimized Defaults F4: Save & Exit ESC: Exit</pre>

5. Ensure that the Network boot option is configured as the primary boot option for the K8s master nodes that are to be used for this cluster.

This is an example of a system that will boot from the network with Slot 1 Port 2 and Slot 2 Port 2.

· · · · · · · · · · · · · · · · · · ·	
Information	Administration - Boot Order 🔅 🧿 🥥
System Information	
Firmware & OS Software	User Administration Directory Groups Boot Order Licensing Key Manager Language Firmware Verification
iLO Federation	C Leasey BIOS
Remote Console & Media	
Power & Thermal	
iLO Dedicated Network Port	Арріу
iLO Shared Network Port	
Remote Support	Company De est Orden
Administration	Server Bool Order
Security	Slot 1 Port 2 : Mellanox Network Adapter - 04:3F:72:D3:FC:D8 (PXE IPv4)
Management	ubuntu
Lifecycle Management	Red Hat Enterprise Linux Generic USB Boot
Number Bystem Configuration	NVMe Drive Port 3A Box 1 Bay 1: NVM Express Controller - 11A0A011TXUR-KCD6XLUL3T84-E28EE38C
Anne (max) Anne (migune)	Slot 1 Port 2: Mellanox Network Adapter - 04/3F/2E/D3F/C/DB (HTTP(S) IPv4) Slot 2 Port 2: Mellanox Network Adapter - 04/3F/2E/D47K2 (HTTP(S) IPv4)
Proclam (CL105 Garro)	Slot 1 Port 2: Mellanox Network Adapter - 04:3F:72:03F:C:D8 (HTTP(5) IPv6)
the check of a line of the second secon	Slot 1 Port 2 : Mellanox Network Adapter - 04:3F:72:D3:FC:DB (PXE IPv6) 🔹
Construction of the second sec	
1	Apply Up Down
No.	
Charlense (Manchanet Colleges)	

 Download a BCM ISO from the <u>Bright download site</u>. Select Base Command Manager 9.2, Ubuntu 20.04, and check the Include NVIDIA DGX A100 Software image checkbox.

	Next Steps
Download	Request a live demo
Product Key	Request a quote
129734-681610-309232-882291-967725	+1 408 300 9448
(e.g. 123456-789012-345678-901234-567890)	info@BrightComputing.co
Version	
Bright Cluster Manager 9.2 🗸	
Architecture	
x86_64/amd64 ∨	
Linux Base Distribution	
Ubuntu 20.04 🗸	
Hardware Vendor	
Generic / Other 🖌	
Additional Features	
□ Include CUDA Packages (for NVIDIA GPUs, approx. 1.3GB)	
□ Include OFED and OPA Packages (for Infiniband and Omnipath, approx. 1.5GB)	
Include NVIDIA DGX A100 software image (approx. 3GB)	
Download Location	
United States (San Jose, CA) 🗸	

7. Burn the ISO to a DVD or to a bootable USB device.

It can also be mounted as virtual media and installed using the BMC. The specific mechanism for the latter will vary by vendor.

- 8. Ensure that the BIOS of the target head node is configured in UEFI mode and that its boot order is configured to boot the media containing the Bright installer image.
- 9. Boot the installation media.

10. At the grub menu, choose Start Bright Cluster Manager Graphical Installer.



11. When the Welcome to Base Command Manager splash screen is displayed, select Start installation.

If the splash screen does not launch, launch it from a terminal. systemctl restart bright-installer-configure



12. Accept the Bright EULA.



13. Accept the OS EULA.

🍀 Bright Computing	Bright Cluster Manager installer	v9.2-4 (ROCKY8u5)
Bright EULA	Bright Computing	
Kernel modules	END LISER I TOENSE AGREEMENT FOR BRIGHT COMPLITING SOFTWARE	<u>^</u>
Hardware info	This end user license agreement including the exhibit attached ("Agreement") is	
Installation source	a legal agreement between you and Bright Computing. Inc. or Bright Computing BV, as annicable ("Bright Computing") and governs your use of the Bright Computing	
Cluster settings	software and materials ("SOFTWARE").	
Workload manager	If you are entering into this Agreement on behalf of a company or other legal entity, you represent that you have the legal authority to bind the entity to	
Network topology	this Agreement, in which case "you" will mean the entity you represent.	
Head node	If you don't have the required authority to accept this Agreement, or if you don't accept all the terms and conditions of this Agreement, do not download,	
Compute nodes	install or use the SOFTWARE.	
BMC configuration	You agree to use the SOFTWARE only for purposes that are permitted by (a) this Agreement, and (b) any applicable law, regulation or generally accepted	
Networks	practices or guidelines in the relevant jurisdictions.	
Head node interfaces	1. License.	
Compute nodes interfaces	1.1 Grant.	
Disk layout	Subject to the terms of this Agreement and payment of applicable fees, Bright Computing hereby grants you a non-exclusive, non-transferable license, without	
Additional software	the right to sublicense, to: (i) install and use the SOFTWARE in accordance with the documentation provided with the SOFTWARE, and (ii) make a single copy of the	
Summary	SOFTWARE solely for backup purposes. No orders are binding until accepted by Bright Computing. All orders accepted are subject to the terms of this	
Deployment	Agreement.	
	1.2 Promotional UTTerings.	
	Bright Computing may, from time to time, offer free or discounted pricing	•
	100 · - 2·	
	Continue remotely Load config Show con	fig Back Next

14. Configure the Kernel modules screen.

Leave the default settings unless otherwise instructed by NVIDIA personnel.

👫 Bright Computing	Bright Cluster I		v9.2-5 (UBUNTU200			
Bright EULA	Kernel modules					
Kernel modules	In order to be able to use all the hardware,	n order to be able to use all the hardware, it is important that the correct set of kernel modules are loaded at boot-				
Hardware info	ime. The hardware in this machine has been probed and the kernel modules listed below were loaded. Under most ircumstances it is not necessary to modify the kernel modules selection, but if you wish to do so, you may add or					
Installation source	remove kernel modules here.					
Cluster settings			€ ⊕			
Workload manager	Name	Parameters	Path			
Network topology	acpi_ipmi		• Ø			
Head node	acpi_power_meter	·	• Ø			
Commute and a	acpi_tad		· Ø			
Compute nodes	aesni_intel		· Ø			
BMC configuration	autofs4		· Ø			
Networks	сср		· 0			
Head node interfaces	cec	-	· Ø			
Compute nodes interfaces	crc32_pclmul		• Ø			
Disk layout	crc8		· Ø			
Summary	crct10dif_pclmul		· Ø			
Deployment	cryptd		· Ø			
	crypto_simd		· Ø			
	drm		· Ø			
	drm_kms_helper		· Ø			
	edac mce amd		· _0			
		Continue remotel	y Show.config Back Nex			

15. Configure the Hardware info screen.

Verify that the target storage device and the cabled host network interfaces are correctly displayed.

👫 Bright Computing			Cluster Manager installer	v9.2-5 (UBUNTU2004	
Bright EULA	На	Hardware info			
Kernel modules	The f	ollowing hardware has been	n detected. If not all hardware has	been recognized, you may go back to the kernel	
Hardware info	modu	modules configuration screen to load extra kernel modules.			
Installation source	e	Туре	Device	Model	
Cluster settings	•	Cd Rom	/dev/cdrom	Virtual DVD-ROM	
Workload manager	0	Keyboard	/dev/input/event3	HP Virtual Keyboard	
Network topology	o	Mouse	/dev/input/mice	HP Virtual Keyboard	
Head node			ens10f0	Network interface [qede]	
Compute nodes			ens10f1	Network interface [qede]	
BMC configuration	0	Network Interfaces	ens1f1np1	Network interface [mlx5_core]	
			ens2f1np1	Network interface [mlx5_core]	
Networks			lo	Network interface []	
Head node interfaces	~	Storage	/dev/nvme0	NVMe disk	
Compute nodes interfaces	· ·	Storage	/dev/nvme0n1	NVMe disk	
		Storage Controllers			
Summary	-			100 m	
Deployment			keyboard:		
Deproyment			/dev/input/event3	HP Virtual Keyboard	
			mouse:		
			/dev/input/mice	HP Virtual Keyboard	
			cdrom:		
			/dev/sr0	iLO Virtual DVD-ROM	
			Contin	ue remotely Show config Back Nexto	

16. Configure the Installation source screen.

Select the appropriate device.

Run the media integrity check if there are doubts about the validity of the installation ISO and there is time allotted to test the integrity.

👫 Bright Computing	Bright Cluster Manager installer	v9.2-5		
Bright EULA	DVD/ISO/USB			
Kernel modules	Select from the list below:			
Hardware info	/dev/sr0 (Bright Install Media)			
Installation source	Run media integrity check			
Cluster settings	Validation of /dev/sr0 successful			
Workload manager				
Network topology				
Head node				
Compute nodes				
BMC configuration				
Networks				
Head node interfaces				
Compute nodes interfaces				
Disk layout				
Deployment				
Deployment				
				_
	Continue remotely	Show config	Back	Next

17. Configure the Cluster settings screen.

Enter information from the site survey. An example site survey is in Appendix A.

🔆 Bright Computing	Bright Cluster Manager Ins		v9.2-5 (UBUNTU200
Bright EULA Kernel modules	General cluster settings		
Hardware info Installation source	BasePod		
Cluster settings	Organization name:		
Workload manager Network topology	Administrator email:		
Compute nodes BMC configuration	Send email to the administrator on first boot Time zone:		
Networks Head node interfaces	(GMT-07:00) America/Los_Angeles		*
Compute nodes interfaces Disk layout	× 172.24.190.7		× •
Summary Deployment	K BREE Leave this field omaty if you intend to use DHCP for external ectwark Search domains:		× *
	x middla com. Leove this field enough if you intensi to use DHCP for external retwork.		× *
		Continue remotely	Show config Back Nex

18. Configure Workload manager screen.

Select None.

After head node installation, K8s will be deployed for container orchestration.

Bright EULA	LIDC		
Kernel modules Hardware info Installation source	A workload management system is highly configured or choose 'None' to prevent co Please select workload manager:	F recommended to run compute jobs. Ple nfiguration.	ase choose the one that should be
Cluster settings Workload manager Network topology	PBS Works	Slurm	IBM Spectrum LSF
Compute nodes BMC configuration Networks	Univa Grid Engine		None
Head node interfaces Compute nodes interfaces Disk iayout Summary	No worklo	ad manager will be configured on first b	oot.
Deployment			

19. Configure Network topology screen.

Select Type1.

In a DGX BasePOD architecture, the cluster nodes are connected to the head node over the internal network, with the head node serving as their default gateway.



20. Configure Head node settings screen.

Enter the Hostname and Administrator password.

👫 Bright Computing	Bright Cluster Manager installer v9.2-5		
Bright EULA	Head node settings		
Kernel modules	Hostname:		
Hardware info	basepod-head1		
Installation source	Administrator pareword		
Cluster settings	Administrator password.		
Workload manager			Ø
Network topology	Confirm administrator password:		
Head node	•••••		Ø)
Compute nodes	Hardware manufacturer:		
BMC configuration	Other		*
Networks			
Head node interfaces			
Compute nodes interfaces			
Disk layout			
Summary			
Deployment			
	Continue remotely Show config	Back	Next In

21. Configure Compute nodes screen.

Set the Number of nodes to 4 $\,$

Set Node digits to 2.

👬 Bright Computing	Bright Cluster Manager installer v	9.2-5 (UBUN	TU2004)
Bright EULA	Compute nodes settings		
Kernel modules	Number of racks:		
Hardware info	1		
Installation source	Number of nodes-		
Cluster settings	A		
Workload manager	4		
Network topology	Node start number:		
Head node	1		
Compute nodes	Node base name:		
BMC configuration	node		
Networks	Node diaits:		
Head node interfaces	2		
Compute nodes interfaces			
Disk layout	Hardware manufacturer:		
Summary	Other		Ŧ
Deployment			
	Continue remotely Show co	nfig Back	Nest

- 22. Configure BMC configuration screen.
 - a. Select Yes for both the Head Node and the Compute Nodes.
 - b. Select IPMI from the BMC network type select lists for both the Head Node and the Compute Nodes.
 - c. Select NO to the DHCP question for both node types.
 - d. Select Yes for Automatically configure BMC when node boots?.
 - e. Select New dedicated network from the To which Ethernet segment is BMC connected? select list.

👫 Bright Computing	Bright Cluster Manager ins	taller v9.2-5 (UBUNTU	
Bright EULA	BMC configuration		
Kernel modules			
Hardware info	Head Node	Compute Nodes	
Installation source	Will head node have IPMI/iDRAC/iLO/CIMC compatible BMCs?	Will compute nodes have IPMI/iDRAC/iLO/CIMC compatible BMCs?	
Cluster settings	🛞 Yes 💿 No	🛞 Yes 💿 No	
Workload manager	BMC network type:	BMC network type:	
Network topology	IPMI +	IPMI	*
Head node			
Compute nodes	Vise DHCP to obtain BMC IP addresses? Vise No	Ves No	
BMC configuration	Automatically configure BMC when node boots?	Automatically configure BMC when node boots?	
Networks	🖲 Yes 💿 No	* Yes 💿 No	
Head node interfaces	To which Ethernet segment is BMC connected?	To which Ethernet segment is BMC connected?	
Compute nodes interfaces	New dedicated network 👻	New dedicated network	*
Disk layout			
Summary			

23. Configure the Networks screens.

a. externalnet

Set the Base IP address, Netmask, Gateway, and Domain name according to the site survey.

🍀 Bright Computing	Bright Cluster Manager installer	
Bright EULA	Networks	
Kernel modules	The following IP networks have been pre-configured. Using the controls below, the networ	rk settings may be altered.
Hardware info	externalnet internalnet iominet	۲
Installation source	Name:	
Cluster settings	externainet	
Workload manager	DHCP	
Network topology	Base IP address:	
Head node	10.130.121.0	
Compute nodes	Notwask	
BMC configuration	256 256 0/04)	~ ~
Networks	233.233.233.0([24])	
Head node interfaces	Gateway:	
Compute nodes interfaces	10.130.121.1	I
Disk layout	Domain name:	
Summary	nvidia.com	
Deployment	MTU:	
	1500	
	Continue remotely Show	w config Back Next

b. internalnet

Set the Base IP address and Netmask according to the site survey.

👫 Bright Computing	Bright Cluster Manager Installer	
Bright EULA Kernel modules	Networks The following IP networks have been pre-configured. Using the controls below, the networks	work settings may be altered.
Hardware info Installation source Cluster settings Workload manager Network topology Head node	externalnet internalnet ipminet Name: internalnet Base IP address: 10.130.122.0	۲
Compute nodes BMC configuration Networks	Netmask: 255.255.255.0(/24) Dynamic range start:	× *
Head node interfaces Compute nodes interfaces Disk layout	10.130.122.160 Dynamic range end: 10.130.122.223	
Summary Deployment	Domain name: eth.cluster	
	Gateway: Optional By default the head node will be used as the default gateway. MTU:	
	Continue remotely Sh	how config Back Next

C. ipminet

Set the Base IP address, Netmask, and Gateway according to the site survey.

👬 Bright Computing	Bright Cluster Manager installer v9.2-5 (I	UBUNTU2004)
Bright EULA Kernel modules	Networks The following IP networks have been pre-configured. Using the controls below, the network settings m	ay be altered.
Hardware info Installation source	externalnet internalnet ipminet Name:	۲
Cluster settings Workload manager Network topology	ipminet Base IP address:	
Head node Compute nodes	10.130.111.64 Netmask: 255.255.255.192(/26)	× *
BMC configuration	Domain name: Ipmi.cluster	
Head node interfaces Compute nodes interfaces Disk layout	Gateway: 10.130.111.65	
Summary Deployment	By default the head node will be used as the default gateway. MTU: 1500	
	Continue remotely Show config	Back Next

- 24. Configure the Head node interfaces screen.
 - Set the IP addresses according to the site survey.

- Bright Lompunng		Bright	Cluster Manager				
Bright EULA	Head node	netwo	rk interface	s			
Kernel modules	field field	meento					۲
Hardware info	Interface		Network		IP address		
Installation source	enslflnpl	× *	internalnet	× *	10.130.122.254	× *	Ō
Cluster settings	eps10f1	× *	externalnet	× -	10 130 121 254	× *	÷
Workload manager	CIIDADIA		externolitet		AUTODIAL ALE OF		-
Network topology	ens10f0	× *	ipminet	× *	10.130.111.126	× *	0
Head node							
Compute nodes							
Compute nodes BMC configuration							,
Compute nodes BMC configuration Networks							,
Compute nodes BMC configuration Networks Head node interfaces							,
Compute nodes BMC configuration Networks Head node interfaces Compute nodes interfaces							,
Compute nodes BMC configuration Networks Head node interfaces Compute nodes interfaces Disk layout							,
Compute nodes BMC configuration Networks Head node interfaces Compute nodes interfaces Disk layout Summary							,
Compute nodes BMC configuration Networks Head node interfaces Compute nodes interfaces Disk layout Summary Deployment							,
Compute nodes BMC configuration Networks Head node interfaces Compute nodes interfaces Disk layout Summary Deployment							,
Compute nodes BMC configuration Networks Head node interfaces Compute nodes interfaces Disk layout Summary Deployment							,
Compute nodes BMC configuration Networks Mead node interfaces Compute nodes interfaces Disk layout Summary Deployment							

- $\label{eq:configure} 25. \ Configure \ the \ {\tt Compute nodes network interfaces screen}.$
 - Set the offset for BOOTIF and ipmi0 to 0.0.0.3.

👫 Bright Computing			Cluster Manager ins			9.2-5 (UBUN	
Bright EULA	Compute r	nodes n	etwork interfa	aces			
Kernel modules							۲
Hardware info	Interface		Network		IP offset		
Installation source	BOOTIF	× *	internalnet	× -	0.0.0.3	× *	亡
Cluster settings	ipmi0	× •	ipminet	× -	b.0.0.3	× -	Ċ
Workload manager							
Network topology	IP addresses for ea addresses. The star	ch of the inter ting address i	faces on all nodes will b in the range is determine	e assigned autom ed by adding the s	atically from a con pecified offset to t	secutive range of he base address	of IP s of the
Head node	network. For examp to be assigned 10.1	ole, a selected	i network base address 1 he second node 10.141.0	10.141.0.0 with an	IP offset of 0.0.0.0) will cause the	first node
Compute nodes							
BMC configuration							
Networks							
Head node interfaces							
Compute nodes interfaces							
Disk layout							
Summary							
Deployment							
				Continue rem	otely Show co	nfig Back	Next

26. Configure the Disk layout screen.

- a. Set Select install drives(s).
- b. Set the Head node disk layout.

👫 Bright Computing	Bright Cluster Manager installer	v9.2-5 (UBUNTU2004)
Bright EULA	Disk lavout	
Kernel modules	Installation drives	
Hardware info	Please select a drive to use for this head node installation. If a software RAID layout was se	lected, then multiple
Installation source	drives should be selected.	
Cluster settings	Select install drive(s):	
Workload manager	✓ /dev/nvme0n1 (NVMe disk)	
Network topology	Disk layouts	
Head node	Please select a layout from the predefined list of node layouts. To view and edit the disk lay	yout, click the edit button
Compute nodes	Delow.	
BMC configuration	One his partition	- 14 0
Networks	one big partition	
Head node interfaces	Compute nodes disk layout:	
Compute nodes interfaces	Default Standard Layout	· Ľ ⊕
Disk layout		
Summary		
Deployment		
	Continue remotely Show	config Back Next

27. Review the Summary screen.

This is a sample Summary screen.

👫 Bright Computing	Bright Clust		v9.2-5		
Bright EULA	Summary				
Kernel modules	Below is a brief summary of some of th	ne installation settings that were selected:			
Hardware info	Drimany outernal interface (D)	10 130 131 0			
Installation source	Primary external interface Netmask: Primary external interface Gateway	24			
Cluster settings	Primary internal interface IP:	10.130.122.0			
Workload manager	Primary internal interface Netmask: Nameservers:	24 8.8.8.8			
Network topology	Timezone: Time servers	America/Los_Angeles 172.24.180.7			
Head node	Workload manager Head node hardware vendor	None Other			
Compute nodes	Compute nodes hardware vendor Install drives	Other /dev/nyme0n1			
BMC configuration	Head node BMC type	IPMI			
Networks	Compute nodes BMC type	1PM1			
Head node interfaces					
Compute nodes interfaces					
Disk layout					
Summary					
Deployment					
		Continue remotely	Show config	Back	Starta

28. Configure the Deployment screen.

- a. Check the Automatically reboot after installation is complete checkbox to reboot the host upon successful completion of the deployment.
- b. Select Install log to see a summary of the installation.



2.1 Cluster Configuration

- Log in to the BCM head node assigned to externalnet. ssh <externalnet>
- 2. Install the cluster license by running the request-license command.

Because HA is used, specify the MAC address of the first NIC of the secondary head node so that it can also serve the Bright licenses in the event of a failover.

This example is for a head node with Internet access. For air-gapped clusters, see "Off-cluster WWW access" in Section 4.3.3 of the <u>Bright Installation manual</u>.

```
# request-license
Product Key (XXXXX-XXXXX-XXXXX-XXXXX-XXXXX): 123456-123456-123456-123456
Country Name (2 letter code): US
State or Province Name (full name): California
Locality Name (e.g. city): Santa Clara
Organization Name (e.g. company): NVIDIA
Organizational Unit Name (e.g. department): MLOPs
Cluster Name: BasePOD
Private key data saved to /cm/local/apps/cmd/etc/cluster.key.new
Warning: Permanently added 'basepod-head1' (ECDSA) to the list of known hosts.
MAC Address of primary head node (basepod-head1) for ens1f1np1
[04:3F:72:E7:67:07]:
Will this cluster use a high-availability setup with 2 head nodes? [y/N] y
```

```
MAC Address of secondary head node for eth0 [XX:XX:XX:XX:XX:XX:XX]:
14:02:EC:DA:AF:18
Certificate request data saved to /cm/local/apps/cmd/etc/cluster.csr.new
Submit certificate request to
http://licensing.brightcomputing.com/licensing/index.cgi ? [Y/n] Y
Contacting http://licensing.brightcomputing.com/licensing/index.cgi...
License granted.
License data was saved to /cm/local/apps/cmd/etc/cluster.pem.new
Install license? [Y/n] Y
====== Certificate Information =======
Version:
                                  7.0 and above
Edition:
                                 Advanced
Common name:
                                 BasePOD
                                NVIDIA
Organization:
                               Sales
Organizational unit:
                                Santa Clara
Locality:
                                California
State:
Country:
                                US
Country.Serial:Starting date:11 Jan 2022Expiration date:07 Jan 2023MAC address / Cloud ID:04:3F:72:E7:67:07|14:02:EC:DA:AF:1832Vac
Licensed nodes with accelerators: 1024
Accounting & Reporting: Yes
Allow edge sites:
                                 Yes
License type:
                                Commercial
_____
Is the license information correct ? [Y/n] Y
```

3. Backup the default software image.

The backup image can be used to create additional software images.

```
# cmsh
% softwareimage
% clone default-image default-image-orig
% commit
```

Note: This document uses # to indicate commands executed as the root user on a head node, and % to indicate commands executed within cmsh. The prompt change is in the preceding block. If it is unclear where a command is being executed, check the prompt that precedes it.

Wait for the ramdisk to be regenerated and the following text to be displayed.

```
Tue Oct 25 16:24:03 2022 [notice] basepod-head1: Initial ramdisk for image default-image-orig was generated successfully
```

4. Backup the DGX software image.

The backup image can be used to create additional software images.

```
% softwareimage
% clone dgx-a100-image dgx-a100-image-orig
% commit.
```

Wait for the ramdisk to be regenerated and the following text to be displayed.

```
Tue Oct 25 16:28:13 2022 [notice] basepod-head1: Initial ramdisk for image dgx-a100-image-orig was generated successfully
```

5. Create the K8s software image by cloning the default software image.

This software image will be further configured and provisioned onto the K8s master nodes.

```
% softwareimage
% clone default-image k8s-master-image
% commit
```

6. Add the required kernel modules to the dgx-a100-image software image.

```
% softwareimage
% use dgx-a100-image
% kernelmodules
% add mlx5_core
% add bonding
% add raid0
% add raid1
% softwareimage commit
```

7. Add the required kernel modules to the k8s-master-image software image.

```
% /
% softwareimage
% use k8s-master-image
% kernelmodules
% add mlx5_core
% add bonding
```

- % softwareimage commit
- 8. Create the dgx node category and assign the dgx-a100-image software image to it.

All nodes assigned to the $\tt dgx$ category will be provisioned with the $\tt dgx-100-image$ software image.

```
% category
% clone default dgx
% set softwareimage dgx-a100-image
% commit
```

9. Create the k8s-master node category and assign the k8s-master-image software image to it.

All nodes assigned to the k8s-master category will be provisioned with the k8s-master-image software image.

```
% category
% clone default k8s-master
% set softwareimage k8s-master-image
% commit
```

10. Create the DGX nodes.

node01 was created during head node installation. Clone node01 to create the DGX nodes, which will initially be named node02, node03, node04, and node05.

```
% device
% foreach --clone node01 -n node02..node05 ()
% commit
```

11. Rename the DGX nodes so they are more easily identified later.

```
% use node02
% set hostname dgx01
% use node03
% use node04
% set hostname dgx03
% use node05
% set hostname dgx04
```

- % device commit
- 12. Clone node01 to create the K8s control plane nodes, which will initially be named

```
node05, node06 and node07.
% device
% foreach --clone node01 -n node06..node08 ()
% commit
```

13. Rename the K8s master nodes so they are more easily identifiable.

```
% device
% use node06
% set hostname knode01
% use node07
% set hostname knode02
% use node08
% set hostname knode03
% device commit
```

14. Rename node01.

The purpose of this step is to specify that node01 is only a template.

```
% device
```

% use node01

```
\% set hostname template01
```

```
% commit
```

15. Assign the DGX nodes to the DGX node category.

% foreach -n dgx01..dgx04 (set category dgx)

16. Assign the K8S nodes to the k8s-master node category.

```
% foreach -n knode01..knode03 (set category k8s-master)
% commit
```

17. Check the nodes and their categories.

Extra options are used for device list to make the format more readable.

```
% device list -f hostname:20,category:10,ip:20,status:15
```

hostname (key)	category	ip	sta	atus	
basepod-head1 dax01	dax	10.130.122.254 10.130.122.5	 [[UP DOWN]
dgx02	dgx	10.130.122.6	[DOWN]
dgx03	dgx	10.130.122.7	[DOWN]
dgx04	dgx	10.130.122.8	[DOWN]
knode01	k8s-master	10.130.122.9	[DOWN]
knode02	k8s-master	10.130.122.10	[DOWN]
knode03	k8s-master	10.130.122.11	[DOWN]
template01	default	10.130.122.4	[DOWN]

2.1.1 Network Configuration

1. Add a Network for InfiniBand (ibnet).

```
% network
% add ibnet
% set domainname ibnet.cluster.local
% set baseaddress 10.149.0.0
% set broadcastaddress 10.149.255.255
% set netmaskbits 16
% set mtu 2048
% commit
```

2. Verify the results.

% list -f name:20,	type:10,netr	maskbits:10,	baseaddress:15,	domainname:20
name (key)	type	netmaskb:	it baseaddress	domainname
externalnet	External	24	10.130.121.0	nvidia.com
globalnet	Global	0	0.0.0.0	cm.cluster
ibnet	Internal	16	10.149.0.0	ibnet.cluster.local
internalnet	Internal	24	10.130.122.0	eth.cluster
ipminet	Internal	26	10.130.111.64	ipmi.cluster

3. Set the IP address of the ipmi0 (BMC) interface.

8 use ipmi0 % set ip 10 130 111 66	
% show	
Parameter	Value
Revision	
Туре	bmc
Network device name	ipmi0
Network	ipminet
IP	10.130.111.66
DHCP	no
Alternative Hostname	
Additional Hostnames	
Start if	always
BringUpDuringInstall	no
On network priority	10
Gateway	0.0.0.0
VLAN ID	0
LAN channel	1

4. Verify that the OS-visible physical interface (enslof) is configured.

This allows the head node to communicate on the ${\tt ipminet}$ network, and therefore, communicate with its own BMC.

% show Parameter 	Value	
Revision Type Network device name Network IP DHCP Alternative Hostname Additional Hostnames	physical ens10f0 ipminet 10.130.111.126 no	

```
Start ifalwaysBringUpDuringInstallnoOn network priority60MAC00:00:00:00:00:00SpeedCard Type
```

5. Add a physical interface if one is not shown in Step 4.

```
% add physical ens10f0
% set network ipminet
% set ip 10.130.111.126
% commit
```

6. Ensure that both head node interfaces are connected to ipminet.

7. Reboot the head node.

A reboot of the head node is required because the network interfaces were changed. $_{\rm \%}$ /

```
% /
% device
% use master
% reboot
Reboot in progress for: basepod-head1
```

2.1.2 Configure Disk Layouts for Node Categories

Part of using BCM for managing nodes in a DGX BasePOD is to define the disk partitions. Each DGX BasePOD node category includes K8s master and DGX node categories, both of which must have their disk layout configured.

2.1.2.1 Configure Disk Layout for the k8s-master Category

1. Augment the disksetup of the k8s-master category.

For the K8s master nodes, an EFI System Partition of 100 MB is created at the start of the disk, with the remainder of the disk dedicated to the OS as a single large partition. Note that this disk setup does not have a swap partition.

The configuration file references /dev/nvme0n1 as the block device used. This may need to be changed to match the specific device name used on systems intended as K8s master nodes.

Save the following text to /cm/local/apps/cmd/etc/htdocs/disk-setup/k8sdisksetup.xml, factoring in any necessary changes specific to the target systems as noted.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<diskSetup xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
<device>
```

```
<blockdev>/dev/nvme0n1</blockdev>
   <partition id="a0" partitiontype="esp">
     <size>100M</size>
     <type>linux</type>
     <filesystem>fat</filesystem>
     <mountPoint>/boot/efi</mountPoint>
     <mountOptions>defaults, noatime, nodiratime</mountOptions>
   </partition>
   <partition id="a1">
     <size>max</size>
     <type>linux</type>
     <filesystem>xfs</filesystem>
     <mountPoint>/</mountPoint>
     <mountOptions>defaults, noatime, nodiratime</mountOptions>
   </partition>
 </device>
</diskSetup>
```

2. Assign this disk layout to the k8s-master node category.

```
$ cmsh
% category
% use k8s-master
% set disksetup /cm/local/apps/cmd/etc/htdocs/disk-setup/k8s-disksetup.xml
% commit
```

2.1.2.2 Configure Disk Layout for the DGX Node Category

The DGX A100 system contains two NVMe drives for a RAID 1 boot volume, and an additional eight NVMe drives that are used as a RAID 0 volume for local, high-performance storage. The following disk configuration creates that disk layout, making use of the <code>/raid</code> directory that was created in the <code>dgx-a100-image</code> software image as the mount point for the local RAID 0 array.

1. Copy the following disk setup XML file to a file on the head node.

```
In this example, it is copied to /cm/local/apps/cmd/etc/htdocs/disk-
setup/dgxa100-disksetup.xml.
<?xml version="1.0" encoding="UTF-8"?>
<diskSetup>
 <device>
   <blockdev>/dev/nvme2n1</blockdev>
    <partition id="boot" partitiontype="esp">
     <size>512M</size>
     <type>linux</type>
     <filesystem>fat</filesystem>
     <mountPoint>/boot/efi</mountPoint>
     <mountOptions>defaults, noatime, nodiratime</mountOptions>
    </partition>
    <partition id="slash1">
     <size>max</size>
     <type>linux raid</type>
   </partition>
  </device>
  <device>
```

```
<blockdev>/dev/nvme3n1</blockdev>
   <partition id="slash2">
     <size>max</size>
     <type>linux raid</type>
   </partition>
 </device>
 <device>
 <blockdev>/dev/nvme0n1</blockdev>
 <partition id="raid1" partitiontype="esp">
   <size>max</size>
   <type>linux raid</type>
 </partition>
</device>
 <device>
 <blockdev>/dev/nvme1n1</blockdev>
 <partition id="raid2" partitiontype="esp">
   <size>max</size>
   <type>linux raid</type>
 </partition>
</device>
<device>
 <blockdev>/dev/nvme4n1</blockdev>
 <partition id="raid3" partitiontype="esp">
   <size>max</size>
   <type>linux raid</type>
 </partition>
</device>
<device>
 <blockdev>/dev/nvme5n1</blockdev>
 <partition id="raid4" partitiontype="esp">
   <size>max</size>
   <type>linux raid</type>
 </partition>
</device>
<device>
 <blockdev>/dev/nvme6n1</blockdev>
 <partition id="raid5" partitiontype="esp">
   <size>max</size>
   <type>linux raid</type>
 </partition>
</device>
<device>
 <blockdev>/dev/nvme7n1</blockdev>
 <partition id="raid6" partitiontype="esp">
   <size>max</size>
   <type>linux raid</type>
 </partition>
</device>
<device>
<blockdev>/dev/nvme8n1</blockdev>
```

```
<partition id="raid7" partitiontype="esp">
    <size>max</size>
    <type>linux raid</type>
  </partition>
 </device>
 <device>
  <blockdev>/dev/nvme9n1</blockdev>
  <partition id="raid8" partitiontype="esp">
    <size>max</size>
    <type>linux raid</type>
  </partition>
 </device>
 <raid id="slash">
  <member>slash1</member>
  <member>slash2</member>
  <level>1</level>
  <filesystem>ext4</filesystem>
  <mountPoint>/</mountPoint>
  <mountOptions>defaults, noatime, nodiratime</mountOptions>
 </raid>
 <raid id="raid">
  <member>raid1</member>
  <member>raid2</member>
  <member>raid3</member>
  <member>raid4</member>
  <member>raid5</member>
  <member>raid6</member>
  <member>raid7</member>
  <member>raid8</member>
  <level>0</level>
  <filesystem>ext4</filesystem>
  <mountPoint>/raid</mountPoint>
  <mountOptions>defaults, noatime, nodiratime</mountOptions>
 </raid>
</diskSetup>
```

- 2. Assign this disksetup to the dgx-a100 node category.
 - \$ cmsh
 % category
 % use dgx

```
% set disksetup /cm/local/apps/cmd/etc/htdocs/disk-setup/dgxa100-disksetup.xml
```

```
% commit
```

2.1.3 Configure Node Network Interfaces

2.1.3.1 Configure Bright to Allow MAC Addresses to PXE Boot

- 1. Use the root (not cmsh) shell.
- 2. In /cm/local/apps/cmd/etc/cmd.conf, uncomment the AdvancedConfig parameter.
 AdvancedConfig = { "DeviceResolveAnyMAC=1" } # modified value
- 3. Restart the CMDaemon to enable reliable PXE booting from bonded interfaces.

```
# systemctl restart cmd
```

The cmsh session will be disconnected because of restarting the CMDaemon. Type connect to reconnect after the CMDaemon has restarted. Or enter exit and then restart cmsh.

2.1.3.2 Configure Provisioning Interfaces on the DGX Nodes

The steps that follow are performed on the head node and should be run on all DGX systems.

Note: Double check the MAC address for each interface, and the IP address for the bond0 interface. Mistakes here will be difficult to diagnose.

1. Add additional physical interfaces and set their MAC addresses.

```
# cmsh
% device
% use dgx01
% interfaces
% add physical enp225s0f1
% set mac B8:CE:F6:2F:08:69
% add physical enp97s0f1
% set mac B8:CE:F6:2D:0E:A7
% add physical enp225s0f1np1
% set mac B8:CE:F6:2F:08:69
% add physical enp97s0f1np1
% set mac B8:CE:F6:2D:0E:A7
% % commit
% list
            Network device name IP
Type
                                                     Network
                                                                       Start if
_____ ____
bmcipmi010.130.111.68ipminetphysicalBOOTIF [prov]10.130.122.5internalnetphysicalenp225s0f10.0.0.0physicalenp97s0f10.0.0.0physicalenp225s0f1np10.0.0.0physicalenp97s0f1np10.0.0.0
                                                                      always
                                                                      always
                                                                       always
                                                                        always
                                                                        always
                                                                        always
```

2. Add a bond0 interface and assign it an IP.
| | Туре | bond |
|----|---|---------------------------|
| | Network device name | bond0 |
| | Network | internalnet |
| | IP | 10.130.122.5 |
| | DHCP | no |
| | Alternative Hostname | |
| | Additional Hostnames | |
| | Start if | always |
| | BringUpDuringInstall | no |
| | On network priority | 70 |
| | Mode | 4 |
| | Options | miimon=100 |
| | Interfaces | enp225s0f1,enp97s0f1 |
| 3. | Remove the bootif interface (bon | d0 will be used instead). |

```
% remove bootif
% exit  # go up two levels
% exit
```

4. Set the provisioninginterface property to bond0.

% set provisioninginterface bond0
% device commit

5. Verify the configuration.

6. Perform all the steps in this section on each of the remaining DGX nodes.

2.1.3.3 Configure Provisioning Interfaces on the K8s Nodes

The steps that follow are performed on the head node and shown for a single K8s node (knode01). But all the steps in this section must be run for each of the three K8s nodes.

1. Add additional physical interfaces and set their MAC addresses.

2. Add a bond0 interface and assign it an IP address.

```
% /
% device
% use knode01
% interfaces
% add bond bond0
% set interfaces enslflnp1 ens2flnp1
% set network internalnet
% set ip 10.130.122.9
% set mode 4
% set options miimon=100
% show
                            Value
Parameter
_____
Revision
Туре
                              bond
                             bond0
Network device name
Network
                             internalnet
ΤP
                             10.130.122.9
DHCP
                              no
Alternative Hostname
Additional Hostnames
Start if
                              always
BringUpDuringInstall
                              no
                              70
On network priority
Mode
                              4
Options
                              miimon=100
Interfaces
                              ens1f1np1,ens2f1np1
```

3. Remove the **bootif** interface and set the provisioning interface.

- 4. Set the provisioninginterface property to bond0.
 % set provisioninginterface bond0
 % device commit
- 5. Perform the steps in this section on the remaining K8s master nodes.

2.1.3.4 Configure InfiniBand Interfaces on DGX Nodes

The following procedure adds four physical InfiniBand interfaces for a single DGX system (dgx01). This procedure must be run for each DGX node.

```
% / # go to top level of CMSH
% device
% use dgx01
% interfaces
% add physical ibp12s0
% set ip 10.149.0.5
% set network ibnet
% add physical ibp75s0
% set ip 10.149.1.5
% set network ibnet
% add physical ibp141s0
% set ip 10.149.2.5
```

```
% device commit
```

2.1.3.5 Identify the Cluster Nodes

1. Identify the nodes by setting the MAC address for the provisioning interface for each node to the MAC address listed in the site survey.

```
% device
% use dgx01
% set mac b8:ce:f6:2f:08:69
% use dgx02
% set mac 0c:42:a1:54:32:a7
% use dgx03
% set mac 0c:42:a1:0a:7a:51
% use dgx04
% set mac 1c:34:da:29:17:6e
% use knode01
% set mac 04:3F:72:E7:64:97
% use knode02
% set mac 04:3F:72:D3:FC:EB
% use knode03
% set mac 04:3F:72:D3:FC:DB
% foreach -c dgx,k8s-master (get mac)
B8:CE:F6:2F:08:69
OC:42:A1:54:32:A7
0C:42:A1:0A:7A:51
1C:34:DA:29:17:6E
04:3F:72:E7:64:97
04:3F:72:D3:FC:EB
04:3F:72:D3:FC:DB
```

- 2. If all the MAC addresses are set properly, commit the changes.
 - % device commit
 - % quit

2.1.3.6 Update the Software Images

These steps update the K8s master nodes and DGX systems to the latest software images.

1. Update K8s master node software image on the BCM head node.

```
# cm-chroot-sw-img /cm/images/k8s-master-image/
/# apt update && apt -y upgrade
/# exit
```

2. Power on dgx01.

Wait until it is recognized by BCM before proceeding.

3. Update the DGX software.

The DGX OS is updated by updating a running DGX node in DGX BasePOD, then using the resultant system as the source for the DGX image.

```
# ssh dgx01
# apt update && apt install cuda-compute-repo && apt full-upgrade -y
# exit
```

4. Set the DGX system to use the dgx-al00-image and set the kernel version to the latest one available.

% device % use dgx01 % grabimage -w % softwareimage % use dgx-a100-image % set kernelversion 5.4.0-131-generic % commit

2.2 Power On and Provision Cluster Nodes

Now that the required post-installation configuration has been completed, it is time to power on and provision the cluster nodes. After the initial provisioning, power control will be available from within Bright—using the cmsh or Bright View. But for this initial provisioning it is necessary to power them on outside of Bright (that is, using the power button or a KVM).

It will take several minutes for the nodes to go through their BIOS. After that, the node status will progress as the nodes are being provisioned. Watch the /var/log/messages and /var/log/node-installer log files to verify that everything is proceeding smoothly.

2.3 Deploy Docker

Install docker on the head node and K8s control plane nodes so that users can use docker functions on those nodes, for example, build containers.

- 1. Run the cm-docker-setup CLI wizard on the head node as the root user.
- # cm-docker-setup
- 2. Choose Deploy to continue.

Docker operations
DeployDocker installation wizardUninstallUninstall DockerExitReturn to the command line
< 0k >

3. By default, the wizard will create a docker configuration overlay. This assigns the Docker::Host role to the nodes selected in the wizard.

Select name for the new configuration overlay Configuration overlay name docker
Configuration overlay name docker
< 0k > < Back >

4. Leave items unselected in the screen because individual nodes will be specified in the next step.

|--|

5. basepod head1, knode01, knode02, and knode03 are selected to install docker.

Select nodes for Do	ocker
<pre>[X] basepod-head1 [] dqx01</pre>	category:dgx
[] dgx02	category:dgx
[] dgx03	category:dgx
[] dgx04	category:dgx
[X] knode01	category:k8s
[X] knode02	category:k8s
[X] Knode03	category: K85
[] temptateor	
	< Ok > < Back >

6. Optionally, specify a specific Docker bridge network. If you choose not to specify a bridge network, the default value of 172.17.0.0/16 will be used.



7. Enter any local Docker repositories this screen.

Registry Drawy for decker			
negistry Floxy for docker			
1			Registry proxies
2 3			
4			
5			
5			
5 6 7 8			
5			
5 0 2 0	< Ok ≫ < Back >	< Help >	
5 0 7 9	<mark>₹ Ok ></mark> < Back >	< Help >	
	<mark>- Ok →</mark> < Back →	< нађр >	
5 0 0 0	<mark>€ OK - S</mark> ≪ Back ≫	< Halp >	
5 0 7 8	Contract of Back 3	K Help s	
5 0 7 9	COCC CRack 3	e Halp a	
5 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	a Back s	« Help's	
5 0 7 0 	singer stark a	K Help s	
5 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	Callonne – Back s	K Help S	
	C C C C C C C C C C C C C C C C C C C	K Helpis	
5 <u>0</u>	Contraction of the second s	K Help S	
5 0 P 0 	Sack >	K Help s	

8. Do not install the NVIDIA Container Runtime on the head node and K8s control plane nodes since there are no GPUs on those nodes.



9. Select Save config & deploy to save the deployment configuration.

Summary Save config & deploy Show config Save config Save config & exit Exit	
<pre>< Ok > < Back ></pre>	

10. By default, the Docker wizard will save the deployment configuration in /root/cm-docker-setup.conf.

This configuration file can be used to redeploy Docker in the future. Select ${\rm ok}$ to start the installation.

ase specify t	he filepath.						
oot/cm-docker	-setup.conf						
rightManuals/ in/ nap/ ests/	4.0K 234 51 25 289						
Show hidden	[]	Resolve symlinks	[] Show details				
				< 0k >	< Back >		

2.4 Deploy K8s

- 1. Run the <code>cm-kubernetes-setup</code> CLI wizard as the root user on the head node. # <code>cm-kubernetes-setup</code>
- 2. Choose Deploy to start the deployment.



3. Select Use existing third party Docker package. The DGX A100 software image comes with Docker.



4. Select Ok to continue.

5. Accept the default settings for this K8s cluster.



6. Select K8s version v.1.21 in the dialog that appears next.

Choose a Kubernetes version. (X) Kubernetes v1.21 () Kubernetes v1.24 < Ok > < Back >	Choose a Kubernetes version. [X] Kubernetes v1.21 () Kubernetes v1.24 Choose a Kubernetes v1.24 Choose a Kubernetes v1.24 Choose a Kubernetes v1.24
Choose a Kubernetes version. (X) Kubernetes v1.21 () Kubernetes v1.24 < Ok > < Back >	Choose a Kubernetes version. (X) Kubernetes v1.21 () Kubernetes v1.24 < Ok > < Back >
(\) Kubernetes v1.21 () Kubernetes v1.24 < Ok > < Back >	() Kubernetes V1.21 () Kubernetes V1.24 < Ok > < Back >

 Select yes to expose the K8s API server on the head node. This allows users to use the K8s cluster from the head node.



8. Select internalnet since the K8s master nodes and the DGX nodes, which are the K8s worker nodes, are all connected on internalnet.



9. Select all three K8s control plane nodes: knode01, knode02, and knode03.

Select master por	les for Kubernetes	
Select master not	les for Rubernetes.	
[] basepod-head	11	
	category:dgx	
[] dgx02	category:dgx	
[] dgx04	category:dgx	
[X] knode01	category:k8s	
[X] knode02	category:k8s	
[X] knode03	category:k8s	
[] temptateor	category:default	
	< 0k > < Back >	

10. Select dgx for the node categories.

Select node categories to use as Kubernet	es workers
[] default	
[X] dgx [] k8s-master	
<pre></pre>	

11. Select all three K8s control plane nodes: knode01, knode02, and knode03 for the Etcd nodes.

Kubernetes Setup		
	Select an odd number of Etcd nodes [] basepod-head1 [] dgx01 category:dgx [] dgx02 category:dgx [] dgx03 category:dgx [] dgx04 category:dgx [] dgx04 category:k8s [X] knode01 category:k8s [X] knode02 category:k8s	
	<pre>[] template01 category:default </pre> < Ok > < Back >	J

12. Accept the defaults unless the organization requires specific ports.

13. Select the Calico network plugin when prompted.



14. Select the following operators: NVIDIA GPU Operator, Prometheus Adapter, Prometheus Adapter Stack, and the cm-jupyter-kernel-operator to install.

[X] NVIDIA GPU Operator [X] Prometheus Adapter [X] Prometheus Operator Stack [X] cm-jupyter-kernel-operator [] cm-kubernetes-postgresql-operator [] cm-kubernetes-spark-operator [] cm-kubernetes-spark-operator	which operators packages to install	
< 0k > K Back >	<pre>[X] NVIDIA GPU Operator [X] Prometheus Adapter [X] Prometheus Operator Stack [X] cm=jupyter-kernel-operator [] cm-kubernetes-postgresql-operator [] cm-kubernetes-spark-operator</pre>	
	< Ok > < Back >	

15. Select the same four operators to be rolled-up with the defaults.



16. Select the Ingress Controller (Nginx), Kubernetes Dashboard, Kubernetes Metrics Server, and Kubernetes State Metrics to deploy.

Which addons do you want to deploy? [X] Ingress Controller (Nginx) [X] Kubernetes Dashboard [X] Kubernetes Metrics Server [X] Kubernetes State Metrics < Ok > < Back >	

17. Select the defaults unless specific ingress ports are to be used.



18. Select no since the K8s control plane nodes do not have GPUs.

Do you wish to install the Bright NVIDIA packages? - cm-nvidia-container-toolkit - cuda-dcgm - cuda-driver These will be installed in the following software images: - /cm/images/k8s-image
yes no

19. Select yes to deploy the Permission Manager.



20. Leave these fields blank, which is the default.

Config	ure permission manager	v (ontional)		
Custom	Custom controller image Custom RBAC Proxy image	e (optional) e (optional) e (optional)		
	<	0k > <	Back >	

21. Select both enabled and default for the Local path storage class.

Configure Kubernetes StorageClass	
StorageClass enabled default	
CEPH [] () CEPH is not available. Local path (X) Do not set default ()	
< 0k > < Back >	

22. Accept the default data storage path and leave the other two fields blank, which is the default.

Configure local path storage pool for Kubernetes
Path to store the data <mark>/cm/shared/apps/kubernetes/default/var/volumes</mark> Custom address of the registry (optional) Custom provisioner's image (optional)
< Ok > < Back >

23. Select Save config & deploy.

Summary	
Save config & deploy Show config Save config Save config & exit Exit	
< 0k > < Back >	

24. Accept the default location for the cm-kubernetes-setup.conf file.

This file can be used to deploy K8s or edited to deploy additional K8s clusters.

rightManuals/	4.0K 234			
in/ m/	51 61			
nap/ ig.xml	25 1.2K			
gxa100-disksetup.xml gxa100-disksetup.xml.orig	2.5K 2.6K			
8s-one-big-partition-xfs.xm ode02_disksetup.xml	l 1.1K 1.8K			

Wait for the installation to finish.

25. Verify that the K8s cluster is installed properly.

```
# module load kubernetes/default/1.21.4
# kubectl cluster-info
Kubernetes control plane is running at https://localhost:10443
CoreDNS is running at https://localhost:10443/api/v1/namespaces/kube-
system/services/kube-dns:dns/proxy
To further debug and diagnose cluster problems, use 'kubectl cluster-info dump'.
# kubectl get nodes
NAME STATUS ROLES
                                        AGE VERSION
knode01 Ready control-plane, master 5mlls v1.21.4
knode02 Ready control-plane, master 5mlls v1.21.4
knode03 Ready control-plane, master 5ml1s v1.21.4
dgx01 Ready worker
dgx02 Ready worker
dgx03 Ready worker
dgx04 Ready worker
                                         5m5s v1.21.4
                                        5m5s v1.21.4
                                        4m54s v1.21.4
                                        4m58s v1.21.4
```

2.4.1 Install the Network Operator

2.4.1.1 Preparation

 Verify that the NVIDIA Mellanox OFED package version on the DGX systems matches the version listed in DGX OS release notes: <u>https://docs.nvidia.com/dgx/dgx-os-release-notes/index.html</u>

```
# cmsh
% device
% pexec -c dgx -j "ofed_info -s"
[dgx01..dgx04]
MLNX_OFED_LINUX-5.4-3.4.0.0:
```

The correct InfiniBand interfaces used in the compute fabric must be identified and their operation status checked. As noted in Table 2, mlx5_0, mlx5_1, mlx5_4, and mlx5_5 are used and should be verified in working condition. Each interface on each node should be state:Active, Physical stat: LinkUp, and Link layer: InfiniBand.

2. Verify that the interfaces are working properly with the following command:

```
[basepod-head1->device]% pexec -c dgx -j "for i in 0 1 4 5; do ibstat -d \
mlx5 ${i} | grep -i \"mlx5 \\|state\\|infiniband\"; done"
[dgx01..dgx04]
CA 'mlx5 0'
                State: Active
                Physical state: LinkUp
                Link layer: InfiniBand
CA 'mlx5 1'
                State: Active
                Physical state: LinkUp
                Link layer: InfiniBand
CA 'mlx5 4'
                State: Active
                Physical state: LinkUp
                Link layer: InfiniBand
CA 'mlx5 5'
                State: Active
                Physical state: LinkUp
                Link layer: InfiniBand
```

3. Check the SRIOV interface status.

The output should include these values:

- NUM_OF_VFS should be set to 8.
- SRIOV_EN should be True(1).
- Link TYPE P1 should be IB(1).

In this example, only the Link_TYPE_P1 is set correctly. The others need to be set in the next step.

```
[basepod-head1->device]% pexec -c dgx -j "for i in 0 1 4 5; do mst start; \
mlxconfig -d /dev/mst/mt4123_pciconf${i} q; done | grep -e \
\"SRIOV_EN\\|LINK_TYPE\\|NUM_OF_VFS\""
[dgx01..dgx04]
        NUM OF VFS
                                              0
         SRIOV EN
                                             False(0)
        LINK TYPE P1
                                             IB(1)
        NUM OF VFS
                                             0
         SRIOV EN
                                             False(0)
         LINK TYPE P1
                                             IB(1)
         NUM OF VFS
                                             0
         SRIOV EN
                                             False(0)
        LINK TYPE P1
                                             IB(1)
        NUM OF VFS
                                             0
         SRIOV EN
                                             False(0)
         LINK TYPE P1
                                             IB(1)
```

4. Enable SRIOV and set NUM OF VFS to 8 for each interface.

```
Since Link TYPE P1 was set correctly, only the two other values are set below.
   [basepod-head1->device]% pexec -c dqx -j "for i in 0 1 4 5; do mst start; \
   mlxconfig -d /dev/mst/mt4123 pciconf${i} -y set SRIOV EN=1 NUM OF VFS=8; done"
   [dqx01..dqx04]
   Starting MST (Mellanox Software Tools) driver set
   Loading MST PCI module - Success
   [warn] mst pciconf is already loaded, skipping
   Create devices
   Unloading MST PCI module (unused) - Success
   Device #1:
   _____
   Device type: ConnectX6
   Name: MCX653105A-HDA_Ax
Description: ConnectX-6 VPI adapter card; HDR IB (200Gb/s) and 200GbE; single-
   port QSFP56; PCIe4.0 x16; tall bracket; ROHS R6
   Device:
              /dev/mst/mt4123 pciconf0
                                               Next Boot
                                                             New
   Configurations:
                                               False(0) True(1)
           SRIOV EN
            NUM OF VFS
                                                              8
                                               0
   Apply new Configuration? (y/n) [n] : y
   Applying... Done!
   -I- Please reboot machine to load new configurations.
   . . . some output omitted . . .
5. Reboot the DGX nodes to load the configuration.
   % reboot -c dqx
6. Wait for the DGX nodes to be UP before continuing to the next step.
   % list -c dgx -f hostname:20,category:10,ip:20,status:10
   hostname (key) category ip
                                                      status
   _____
   dgx01dgx10.130.122.5[UP]dgx02dgx10.130.122.6[UP]dgx03dgx10.130.122.7[UP]dgx04dgx10.130.122.8[UP]
7. Configure eight SRIOV VFs on the InfiniBand ports.
```

[basepod-head1->device]% pexec -c dgx -j "for i in 0 1 4 5; do echo 8 > \
/sys/class/infiniband/mlx5_\${i}/device/sriov_numvfs; done"

2.4.1.2 Deploy the Network Operator

Perform these steps on the primary head node.

- 1. Load the Kubernetes environment module.
 # module load kubernetes/default/1.21.4
- 2. Add and install the Network Operator Helm repo. # helm repo add nvidia-networking https://mellanox.github.io/network-operator "nvidia-networking" has been added to your repositories

```
# helm repo update
Hang tight while we grab the latest from your chart repositories...
...Successfully got an update from the "nvidia-networking" chart repository
...Successfully got an update from the "prometheus-community" chart repository
...Successfully got an update from the "nvidia" chart repository
Update Complete. #Happy Helming!#
```

3. Create values.yaml file for Helm to install Network Operator.

```
root@basepod-head1:~# cat ./network-operator/values.yaml
nfd:
  enabled: true
sriovNetworkOperator:
  enabled: true
# NicClusterPolicy CR values:
deployCR: true
ofedDriver:
 deploy: false
rdmaSharedDevicePlugin:
 deploy: false
sriovDevicePlugin:
 deploy: false
secondaryNetwork:
  deploy: true
 multus:
   deploy: true
  cniPlugins:
   deploy: true
 ipamPlugin:
    deploy: true
root@basepod-head1:~#
```

4. Use Helm to install Network Operator.

```
# helm install -f ./network-operator/values.yaml -n \
network-operator --create-namespace --wait nvidia-networking/network-operator --generate-
name
NAME: network-operator-1665598416
LAST DEPLOYED: Wed Oct 12 11:13:42 2022
NAMESPACE: network-operator
STATUS: deployed
REVISION: 1
TEST SUITE: None
NOTES:
Get Network Operator deployed resources by running the following commands:
# kubectl -n network-operator get pods
NAME
                                                  READY STATUS RESTARTS AGE
                                                             1/1 Running 0 96s
cni-plugins-ds-9bv8j
                                                              1/1 Running 0 96s
cni-plugins-ds-f87tg
cni-plugins-ds-s76pw
                                                              1/1 Running 0 96s
```

cni-plugins-ds-w2lhl	1/1	Running	0	96s
kube-multus-ds-9gw52	1/1	Running	0	96s
kube-multus-ds-kbkvf	1/1	Running	0	96s
kube-multus-ds-s2n8h	1/1	Running	0	96s
kube-multus-ds-t8hfs	1/1	Running	0	96s
network-operator-1668351491-b58d45f44-qkn7v	1/1	Running	0	102s
network-operator-1668351491-node-feature-discovery-master-5fgrv	1/1	Running	0	102s
network-operator-1668351491-node-feature-discovery-worker-2vnjp	1/1	Running	0	102s
network-operator-1668351491-node-feature-discovery-worker-77pnc	1/1	Running	0	102s
network-operator-1668351491-node-feature-discovery-worker-8pc2k	1/1	Running	0	102s
network-operator-1668351491-node-feature-discovery-worker-17r4r	1/1	Running	0	102s
network-operator-1668351491-sriov-network-operator-5675d88rpqhn	1/1	Running	0	102s
sriov-network-config-daemon-14ch9	3/3	Running	0	83s
sriov-network-config-daemon-199rq	3/3	Running	0	83s
sriov-network-config-daemon-zvcpv	3/3	Running	0	83s
sriov-network-config-daemon-zvsxq	3/3	Running	0	83s
whereabouts-g51kv	1/1	Running	0	96s
whereabouts-hd4zt	1/1	Running	0	96s
whereabouts-jjmcv	1/1	Running	0	96s
whereabouts-kll9p	1/1	Running	0	96s
root@basepod-head1:/nfs/general/network-operator#				

5. Create the sriov-ib-network-node-policy.yaml and sriovibnetwork.yaml configuration files.

```
The interface names must match the interface names in Table 2.
```

```
root@basepod-head1:~# cat ./network-operator/sriov-ib-network-node-policy.yaml
apiVersion: sriovnetwork.openshift.io/v1
kind: SriovNetworkNodePolicy
metadata:
 name: ibp12s0
 namespace: network-operator
spec:
 deviceType: netdevice
 nodeSelector:
   feature.node.kubernetes.io/network-sriov.capable: "true"
 nicSelector:
   vendor: "15b3"
    pfNames: ["ibp12s0"]
  linkType: ib
 isRdma: true
 numVfs: 8
 priority: 90
 resourceName: resibp12s0
apiVersion: sriovnetwork.openshift.io/v1
kind: SriovNetworkNodePolicy
metadata:
name: ibp75s0
 namespace: network-operator
spec:
 deviceType: netdevice
 nodeSelector:
   feature.node.kubernetes.io/network-sriov.capable: "true"
  nicSelector:
   vendor: "15b3"
   pfNames: ["ibp75s0"]
  linkType: ib
 isRdma: true
```

```
numVfs: 8
  priority: 90
  resourceName: resibp75s0
apiVersion: sriovnetwork.openshift.io/v1
kind: SriovNetworkNodePolicy
metadata:
 name: ibp141s0
 namespace: network-operator
spec:
 deviceType: netdevice
 nodeSelector:
   feature.node.kubernetes.io/network-sriov.capable: "true"
 nicSelector:
   vendor: "15b3"
   pfNames: ["ibp141s0"]
 linkType: ib
 isRdma: true
 numVfs: 8
  priority: 90
 resourceName: resibp141s0
apiVersion: sriovnetwork.openshift.io/v1
kind: SriovNetworkNodePolicy
metadata:
 name: ibp186s0
 namespace: network-operator
spec:
 deviceType: netdevice
 nodeSelector:
   feature.node.kubernetes.io/network-sriov.capable: "true"
 nicSelector:
   vendor: "15b3"
    pfNames: ["ibp186s0"]
  linkType: ib
  isRdma: true
  numVfs: 8
 priority: 90
 resourceName: resibp186s0
root@basepod-head1:~# cat ./network-operator/sriovibnetwork.yaml
apiVersion: sriovnetwork.openshift.io/v1
kind: SriovIBNetwork
metadata:
 name: ibp12s0
 namespace: network-operator
spec:
  ipam: |
    {
      "type": "whereabouts",
      "datastore": "kubernetes",
      "kubernetes": {
        "kubeconfig": "/etc/cni/net.d/whereabouts.d/whereabouts.kubeconfig"
      },
      "range": "192.168.1.0/24",
```

```
"log file": "/var/log/whereabouts.log",
      "log level": "info"
   }
  resourceName: resibp12s0
  linkState: enable
 networkNamespace: default
___
apiVersion: sriovnetwork.openshift.io/v1
kind: SriovIBNetwork
metadata:
 name: ibp75s0
  namespace: network-operator
spec:
  ipam: |
    {
      "type": "whereabouts",
      "datastore": "kubernetes",
      "kubernetes": {
       "kubeconfig": "/etc/cni/net.d/whereabouts.d/whereabouts.kubeconfig"
      },
      "range": "192.168.2.0/24",
      "log_file": "/var/log/whereabouts.log",
      "log level": "info"
    }
  resourceName: resibp75s0
  linkState: enable
  networkNamespace: default
apiVersion: sriovnetwork.openshift.io/v1
kind: SriovIBNetwork
metadata:
 name: ibpi141s0
 namespace: network-operator
spec:
  ipam: |
   {
      "type": "whereabouts",
      "datastore": "kubernetes",
      "kubernetes": {
       "kubeconfig": "/etc/cni/net.d/whereabouts.d/whereabouts.kubeconfig"
      },
      "range": "192.168.3.0/24",
      "log file": "/var/log/whereabouts.log",
      "log level": "info"
    }
  resourceName: resibp141s0
  linkState: enable
  networkNamespace: default
apiVersion: sriovnetwork.openshift.io/v1
kind: SriovIBNetwork
metadata:
 name: ibp186s0
namespace: network-operator
```

```
spec:
  ipam: |
    {
      "type": "whereabouts",
      "datastore": "kubernetes",
      "kubernetes": {
        "kubeconfig": "/etc/cni/net.d/whereabouts.d/whereabouts.kubeconfig"
      },
      "range": "192.168.4.0/24",
      "log file": "/var/log/whereabouts.log",
      "log level": "info"
    }
  resourceName: resibp186s0
  linkState: enable
  networkNamespace: default
root@basepod-head1:~#
```

6. Deploy the configuration files.

```
# kubectl apply -f ./network-operator/sriov-ib-network-node-policy.yaml
sriovnetworknodepolicy.sriovnetwork.openshift.io/ibp12s0 created
sriovnetworknodepolicy.sriovnetwork.openshift.io/ibp141s0 created
sriovnetworknodepolicy.sriovnetwork.openshift.io/ibp141s0 created
sriovnetworknodepolicy.sriovnetwork.openshift.io/ibp186s0 created
```

```
# kubectl apply -f ./network-operator/sriovibnetwork.yaml
sriovibnetwork.sriovnetwork.openshift.io/ibp12s0 created
sriovibnetwork.sriovnetwork.openshift.io/ibp75s0 created
sriovibnetwork.sriovnetwork.openshift.io/ibp141s0 created
sriovibnetwork.sriovnetwork.openshift.io/ibp186s0 created
```

7. Deploy the mpi-operator.

```
# kubectl apply -f \
https://raw.githubusercontent.com/kubeflow/mpi-\
operator/master/deploy/v2beta1/mpi-operator.yaml
namespace/mpi-operator created
customresourcedefinition.apiextensions.k8s.io/mpijobs.kubeflow.org
created
serviceaccount/mpi-operator created
clusterrole.rbac.authorization.k8s.io/kubeflow-mpijobs-admin created
clusterrole.rbac.authorization.k8s.io/kubeflow-mpijobs-edit created
clusterrole.rbac.authorization.k8s.io/kubeflow-mpijobs-view created
clusterrole.rbac.authorization.k8s.io/kubeflow-mpijobs-view created
clusterrole.rbac.authorization.k8s.io/mpi-operator created
clusterrole.rbac.authorization.k8s.io/mpi-operator created
clusterrole.rbac.authorization.k8s.io/mpi-operator created
```

8. Copy the Network Operator /opt/cni/bin directory to /cm/shared, where it will be accessed by the head nodes.

```
# ssh dgx01
# cp -r /opt/cni/bin /cm/shared/dgx_opt_cni_bin
# exit
```

9. Make two backup directories.

```
root@basepod-head1:~ # mv /cm/images/dgx-a100-\
image/cm/local/apps/kubernetes/current/bin/cni /cm/images/dgx-a100-\
image/cm/local/apps/kubernetes/current/bin/cni_orig
root@basepod-head1:~# mv /cm/images/dgx-a100-image/opt/cni/bin \
/cm/images/dgx-a100-image/opt/cni/bin orig
```

10. Copy the directory from /cm/shared directory to /cm/image where it is used by the BCM DGX image.

```
root@basepod-head1:~ # cp -r /cm/shared/dgx_opt_cni_bin \
/cm/images/dgx-a100-image/cm/local/apps/kubernetes/current/bin/cni
# cp -r /cm/shared/dgx_opt_cni_bin /cm/images/dgx-a100-image/opt/cni/bin
```

11. Create the network-validation.yaml file and run a simple validation test.

```
root@basepod-head1:~ # cat network-operator/network-validation.yaml
apiVersion: v1
kind: Pod
metadata:
 name: network-validation-pod
  annotations:
   k8s.v1.cni.cncf.io/networks: ibp75s0,ibp186s0,ibp12s0,ibp141s0
spec:
  containers:
    - name: network-validation-pod
      image: docker.io/deepops/nccl-tests:latest
      imagePullPolicy: IfNotPresent
      command:
       - sh
        - -c
        - sleep inf
      securityContext:
        capabilities:
          add: [ "IPC LOCK" ]
      resources:
        requests:
          nvidia.com/resibp75s0: "1"
          nvidia.com/resibp186s0: "1"
          nvidia.com/resibp12s0: "1"
          nvidia.com/resibp141s0: "1"
        limits:
          nvidia.com/resibp75s0: "1"
          nvidia.com/resibp186s0: "1"
          nvidia.com/resibp12s0: "1"
          nvidia.com/resibp141s0: "1"
root@basepod-head1:~ # kubectl apply -f \
./network-operator/network-validation.yaml
pod/network-validation-pod created
```

2.4.1.3 Run a Multi-node NCCL Test

The NVIDIA Collective Communication Library (NCCL) implements multi-GPU and multinode communication primitives optimized for NVIDIA GPUs and networking that is the foundation for many AI/ML training and deep learning applications. A successful run of a multi-node <u>NCCL test</u> is a good indicator that multi-node MPI and NCCL communication between GPUs is operating correctly.

```
1. Build the nccl test K8s job file.
   # cat ./nccl test.yaml
   apiVersion: kubeflow.org/v2beta1
   kind: MPIJob
   metadata:
     name: nccltest
   spec:
     slotsPerWorker: 8
     runPolicy:
      cleanPodPolicy: Running
     mpiReplicaSpecs:
       Launcher:
         replicas: 1
         template:
            spec:
              containers:
              - image: docker.io/deepops/nccl-tests:latest
                name: nccltest
                imagePullPolicy: IfNotPresent
                command:
                - sh
                - "-c"
                - |
                  /bin/bash << 'EOF'
                  mpirun --allow-run-as-root \
                    -np 16 \
                    -bind-to none -map-by slot \
                    -x NCCL DEBUG=INFO \
                    -x NCCL DEBUG SUBSYS=NET \
                    -x NCCL ALGO=RING \setminus
                    -x NCCL_IB DISABLE=0 \
                    -x LD LIBRARY PATH \
                    -x PATH \
                    -mca pml ob1 \
                    -mca btl self,tcp \
                    -mca btl tcp if include 192.168.0.0/16 \
                     -mca oob_tcp_if_include 172.29.0.0/16 \
                    /nccl_tests/build/all_reduce_perf -b 8 -e 4G -f2 -g 1 \
                    && sleep infinity
                  EOF
       Worker:
         replicas: 2
         template:
           metadata:
             annotations:
               k8s.v1.cni.cncf.io/networks: ibp12s0,ibp75s0,ibp141s0,ibp186s0
           spec:
```

```
containers:
```

```
- image: docker.io/deepops/nccl-tests:latest
name: nccltest
imagePullPolicy: IfNotPresent
securityContext:
    capabilities:
    add: [ "IPC_LOCK" ]
resources:
    limits:
    nvidia.com/resibp12s0: "1"
    nvidia.com/resibp141s0: "1"
    nvidia.com/resibp186s0: "1"
    nvidia.com/gpu: 8
```

2. Run the nccl test file.

A sample logfile follows.

kubectl logs -f nccltest-launcher-9pp28

•••

#							out-of-	place			in-p	lace	
#	size	count	type	redop	root	time	algbw	busbw	#wrong	time	algbw	busbw	#wrong
#	(B)	(elements)				(us)	(GB/s)	(GB/s)		(us)	(GB/s)	(GB/s)	
	8	2	float	sum	-1	41.28	0.00	0.00	0	40.99	0.00	0.00	0
	16	4	float	sum	-1	38.58	0.00	0.00	0	40.33	0.00	0.00	0
	32	8	float	sum	-1	40.15	0.00	0.00	0	39.73	0.00	0.00	0
	64	16	float	sum	-1	39.11	0.00	0.00	0	38.47	0.00	0.00	0
	128	32	float	sum	-1	39.66	0.00	0.01	0	40.57	0.00	0.01	0
	256	64	float	sum	-1	39.52	0.01	0.01	0	39.50	0.01	0.01	0
	512	128	float	sum	-1	41.58	0.01	0.02	0	40.09	0.01	0.02	0
	1024	256	float	sum	-1	42.30	0.02	0.05	0	41.71	0.02	0.05	0
	2048	512	float	sum	-1	45.81	0.04	0.08	0	44.70	0.05	0.09	0
	4096	1024	float	sum	-1	48.77	0.08	0.16	0	49.13	0.08	0.16	0
	8192	2048	float	sum	-1	55.33	0.15	0.28	0	55.52	0.15	0.28	0
	16384	4096	float	sum	-1	58.32	0.28	0.53	0	56.24	0.29	0.55	0
	32768	8192	float	sum	-1	60.59	0.54	1.01	0	58.35	0.56	1.05	0
	65536	16384	float	sum	-1	71.26	0.92	1.72	0	71.29	0.92	1.72	0
	131072	32768	float	sum	-1	93.71	1.40	2.62	0	103.5	1.27	2.38	0
	262144	65536	float	sum	-1	103.5	2.53	4.75	0	101.0	2.60	4.87	0
	524288	131072	float	sum	-1	137.8	3.80	7.13	0	148.8	3.52	6.60	0
	1048576	262144	float	sum	-1	107.9	9.72	18.22	0	109.6	9.57	17.94	0
	2097152	524288	float	sum	-1	120.7	17.38	32.59	0	120.9	17.34	32.51	0
	4194304	1048576	float	sum	-1	153.4	27.34	51.26	0	153.3	27.36	51.31	0
	8388608	2097152	float	sum	-1	230.4	36.40	68.26	0	229.6	36.54	68.50	0
	16777216	4194304	float	sum	-1	403.4	41.59	77.98	0	405.1	41.42	77.66	0
	33554432	8388608	float	sum	-1	735.5	45.62	85.54	0	736.4	45.57	85.44	0
	67108864	16777216	float	sum	-1	1426.1	47.06	88.23	0	1459.4	45.98	86.22	0
1	34217728	33554432	float	sum	-1	2639.9	50.84	95.33	0	2682.3	50.04	93.82	0
2	68435456	67108864	float	sum	-1	5202.4	51.60	96.75	0	5187.5	51.75	97.03	0
5	36870912	134217728	float	sum	-1	10469	51.28	96.15	0	10327	51.99	97.48	0
10	73741824	268435456	float	sum	-1	20588	52.15	97.79	0	20786	51.66	96.86	0
21	47483648	536870912	float	sum	-1	41240	52.07	97.64	0	41165	52.17	97.81	0
42	94967296	1073741824	float	sum	-1	82270	52.21	97.89	0	82325	52.17	97.82	0
# Ou	t of bound	s values : 0 OK											
# Av	g bus band	width : 34.00	031										

2.5 (Optional) Deploy Bright Jupyter

BCM provides a robust and popular Jupyter integration. Because the Bright Jupyter integration distributes the kernel across the cluster through the HPC workload management system or Kubernetes, Jupyter is generally installed on the head node or on a login node.

2.5.1 Install Jupyter Using the CLI Wizard

- 1. Run the cm-jupyter-setup CLI wizard on the head node as the root user. ${}_{\#}$ cm-jupyter-setup
- 2. Choose Deploy to continue.

Jupyter operations

3. Specify the overlay name and priority for the JupyterHub login nodes.

By default, the Jupyter wizard will create a configuration overlay named jupyterhub with a priority of 500. Use the defaults unless there is an existing jupyterhub overlay.

4. Select basepod-head1 and then Ok.

After HA is configured, the cm-jupyterhub service will be set to always run on the active head node.

[] dgx01 category:dgx [] dgx02 category:dgx	
[] dgx04 category:dgx	
[] knode01 category:k8s	
[] knode02 category:k85 [] knode03 category:k8s	
[] template01 category:default	
<mark>< 0k ></mark> < Back >	

5. Select the default ports of 8000, 8901, and 8902 and select Ok. Users will access it on the active head node on port 8000.



6. Select Save config & deploy and then Ok.

Summary	
Save config & deploy Show config Advanced settings Save config & exit Exit exit exit	
	l

7. Select Ok to start the installation.

By default, the Jupyter wizard will save the deployment configuration in /root/cm-jupyter-setup.conf. This configuration file can be used to redeploy Jupyter in the future.

ase specify the filepath.				
oot/cm-jupyter-setup.conf				
rightManuals/ in/ map/ ig.xml =-docker.setup.conf =-kubernetes.setup.conf =-kubernetes.setup.conf =-kubernetes.setup.conf dgsa100-disksetup.xml dso-one-big-parition-xfs.xml ode02_disksetup.xml	4.0% 234 254 25 25 1.2% 1.4% 2.6% 2.6% 1.1% 1.1% 1.9% 381			
] show hidden [] R	lesolve symlinks [] Show detai	ls characteristic and characteristic		

8. When the installation completes, the cm-jupyter service will automatically be started on the selected node.

All users in the cluster (except the root user) will be able to login to Jupyterhub using a web browser at http://<head-node-ip or FQDN>:8000.

Examp	ble	e: <u>http://10.13</u>	0.	121.254:800	<u>)0</u>
	\$	File Edit View Run Kernel	Tabs	Settings Help	
		KERNEL TEMPLATES		🖾 Launcher	
	0 ≡	Python on Kubernetes Operator Python+Spark on Kubernetes Op Julia on Kubernetes Operator Python+NGC on Kubernetes Ope KERNEL DEFINITIONS	+ + + +	Not	zbook
	*	WLM CLUSTERS	_	Python	3
		default	¢	S_ Con Python S_ Oth Termina	er I Text File Markdown File Python File Show Contextual Help
	S	imple 🔵 0 🛐 0 🥮			

9. If needed, a test user can be created with following command:
 # cmsh -c "user; add jupyterhubuser; set password jupyterhubuser; commit"

10. Add the user to K8s.

cm-kubernetes-setup --add-user jupyterhubuser --operators cm-jupyter-kerneloperator

Chapter 3. High Availability

This section covers how to configure high availability (HA) using cmha-setup CLI wizard.

1. Ensure that both head nodes are licensed.

We provided the MAC address for the secondary head when we installed the cluster license (section 3.2.2.1). % main licenseinfo | grep ^MAC

MAC address / Cloud ID

04:3F:72:E7:67:07|14:02:EC:DA:AF:18

2. Configure the NFS shared storage.

Mounts configured in fsmounts will be automatically mounted by the CMDaemon.

% device					
use master					
fsmounts					
add /nfs/general					
s set device 10.130.122.252:/var/nfs/general					
🛿 set filesystem nfs					
t commit					
% show					
Parameter	Value				
Device	10.130.122.252:/var/nfs/general				
Revision					
Filesystem	nfs				
Mountpoint	/nfs/general				
Dump	no				
RDMA	no				
Filesystem Check	NONE				
Mount options	defaults				

3. Verify that the shared storage is mounted.

```
# mount | grep '/nfs/general'
10.130.122.252:/var/nfs/general on /nfs/general type nfs4
(rw,relatime,vers=4.2,rsize=1048576,wsize=1048576,namlen=255,hard,proto=tcp,timeo
=600,retrans=2,sec=sys,clientaddr=10.130.122.254,local_lock=none,addr=10.130.122.
252)
```
4. Verify that head node has power control over the cluster nodes.

```
% device
% power -c dgx,k8s-master status
[basepod-head1->device]% power -c dgx,k8s-master status
ipmi0 ..... [ ON ] dgx01
ipmi0 ..... [ ON ] dgx02
ipmi0 ..... [ ON ] dgx03
ipmi0 ..... [ ON ] dgx04
ipmi0 ..... [ ON ] knode01
ipmi0 ..... [ ON ] knode01
ipmi0 ..... [ ON ] knode02
ipmi0 ..... [ ON ] knode03
[basepod-head1->device]%
```

5. Power off the cluster nodes.

The cluster nodes must be powered off before configuring HA.

% power -c k8s-master,dgx off ipmi0 [OFF] knode01 ipmi0 [OFF] knode02 ipmi0 [OFF] knode03 ipmi0 [OFF] dgx01 ipmi0 [OFF] dgx02 ipmi0 [OFF] dgx03 ipmi0 [OFF] dgx04

- 6. Start the cmha-setup CLI wizard as the root user on the primary head node. # cmha-setup
- 7. Select Setup.



8. Select Configure.



9. Verify that the cluster license information found by the wizard is correct.

The following MAC addresses have been found in the license information: 04:3F:72:E7:67:07 14:02:EC:DA:AF:18 If they are correct, then please press 'Continue'. If not, one of the following has to be done: 1. If you have not activated your Product Key, please run request-license and follow instructions. 2. If you have run out of licenses, please contact your reseller, or contact our support. Press 'BACK' to go back to the failover setup menu. 88% - 88% -		
If they are correct, then please press 'Continue'. If not, one of the following has to be done: 1. If you have not activated your Product Key, please run request-license and follow instructions. 2. If you have run out of licenses, please contact your reseller, or contact our support. Press 'BACK' to go back to the failover setup menu. CONTINUES < BACK >	INE TOLLOWING MAL addresses have been found in the licens	e information:
If not, one of the following has to be done: 1. If you have not activated your Product Key, please run request-license and follow instructions. 2. If you have run out of licenses, please contact your reseller, or contact our support. Press 'BACK' to go back to the failover setup menu. CONTINUES < BACK > 88%	If they are correct, then please press 'Continue'.	
<pre>1. If you have not activated your Product Key, please run request-license and follow instructions. 2. If you have run out of licenses, please contact your reseller, or contact our support. Press 'BACK' to go back to the failover setup menu. CONTINUES < BACK > 88% - CONTINUES < BACK ></pre>	If not, one of the following has to be done:	
Press 'BACK' to go back to the failover setup menu.	 If you have not activated your Product Key, please run request-license and follow instructions. If you have run out of licenses, please contact your reseller, or contact our support. 	
CONTINUE> < BACK >	Press 'BACK' to go back to the failover setup menu.	88% -
	CONTINUE> < BACK >	

10. Configure an external virtual IP address that will be used by the active head node in the HA configuration. (This will be the IP that should always be used for accessing the active head nodes.)

			1
Please enter the values IP address must not be a	for the shared ext lready in use or a	secoal interface plastinged to another	raneters. the r interface.
Name: IP [10.100.121.0/24]:	n.18[1.col.c 0.130.121.251		
	- SKIP :-	-C BACK :-	

11. Provide an internal virtual IP address that will be used by the active head node in the HA configuration.

Please enter the values IP address must be in ra be one that is not in us Name: IP [10.130.122.0/24]:	for the shared inte nge of the managemen e already. msifinpi.comha 0.130.122.251	rnal interface para nt network. The int	meters. The erface must
< <mark>Next ></mark>	< SKIP >	< BACK >	-

12. Provide the name of the secondary head node.

Please enter the hostname of the secondary head node. Name: basepod-head2
< NEXT > < BACK >

13. DGX BasePOD uses the internal network as the failover network, so select SKIP to continue.

14. Configure the IP addresses for the secondary head node.



15. The wizard shows a summary of the information that it has collected. This screen shoes the VIP that will be assigned to the internal and external interfaces.

SUMMARY	1	
Failover Setup Summary		
Shared Internal Interface: Name: ensifinp1:cmh IP: 10.130.122.25	1	
Shared External Interface: Name: ens10f1:cmha IP: 10.130.121.25		
Failover Network: SKIPPING		
ı(+) < € <u>XIT ></u>	52% -	

16. Select Yes to proceed with the failover configuration.



17. Enter the MySQL root password. This should be the same as the root password.



18. The wizard implements the first steps in the HA configuration. If all the steps show OK, press ENTER to continue. The progress is shown below:

,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,	-		
Initializing failover setup on master	[OK]
Updating shared internal interface	[OK]
Updating shared external interface	[OK]
Updating extra shared internal interfaces	[OK]
Cloning head node	[OK]
Updating secondary master interfaces	[OK]
Updating Failover Object	[OK]
Restarting cmdaemon	[OK]
Press any key to continue			

19. Run the /cm/cm-clone-install -failover command on the secondary head node. Note that this should be a one-time network boot.

The failover setup initialization on the primary master is done. Now boot the secondary master into the rescue environment and ru	n the
/cm/cm-clone-installfailover	
and follow the instructions.	
Once the installation has begun, select 'Install Progress' from Failover setup menu, to see the installation progress of the clone machine. When the install complete, and the secondary master is up, select 'Finalize' from the Failover setu to complete the failover setup process.	the ation is p menu,
<mark>< 0</mark> K >	94% -

20. PXE boot the secondary head node, then select RESCUE from the grub menu. Since this is the initial boot of this node, it must be done outside of Base Command Manager (BMC or physical power button).

	Olustan Manager BVP Paulaneerst
	Cluster Manager FXE Environment
AUTO	- Normal node boot
MAIN	 Drop to maintenance shell
RESCUE	– Start rescue environment
	Press [Tab] to edit options

21. Once the secondary head node has booted into the rescue environment, run the /cm/cm-clone-install -failover command, then enter yes when prompted. The secondary head node will be cloned from the primary.

22. When cloning is completed, enter $_{\rm Y}$ to reboot the secondary head node. The secondary needs to boot from its hard drive. PXE boot should not be enabled.



- 23. Wait for the secondary head node to reboot and then continue the HA setup procedure on the primary head node.
- 24. Select finalize from the cmha-setup menu.

This will clone the MySQL database from the primary to the secondary head node.

Select 'Configure' to configur	e failover setup, 'Finalize' to finalize
the failover setup if the seco	ndary head node has been installed. Choose
'Clone Install' to see the ins	tallation instructions for the secondary
head node, if the failover con	figuration has been completed. Choose
'Install Progress', if the sec	ondary head node is being installed. Choose
'Undo Failover' to remove exis	ting failover configuration.
Configure	Configure failover setup
Clone Install	View install instructions
Install Progres	View install progress
Toplics	Finalize failover setup
Undo Fallover	Remove failover setup
Main	Main menu
⊾ ■ NEXT >	< BACK >

25. Select <CONTINUE> on the confirmation screen.



26. Enter the MySQL root password. This should be the same as the root password.



- Preparing finalize failover 100%
- 27. The cmha-setup wizard continues. Press ENTER to continue when prompted.

The progress is shown below:

Updating secondary master mac address	[OK]
Initializing failover setup on basepod-head2	[OK]
Stopping cmdaemon	[OK]
Cloning cmdaemon database	[OK]
Checking database consistency	[OK]
Starting cmdaemon, chkconfig services	[OK]
Cloning workload manager databases	[OK]
Cloning additional databases	[OK]
Update DB permissions	[OK]
Checking for dedicated failover network	[OK]
Press any key to continue			

28. The Finalize step is now completed. Select <REBOOT> and wait for the secondary head node to reboot.



29. The secondary head node is now UP.

% device list -f hostname:20,category:12,ip:20,status:15

hostname (key)	category	ip	sta	tus	
basepod-head1		10.130.122.254	[UP]
basepod-head2		10.130.122.253	[UP]
knode01	k8s-master	10.130.122.9	[DOWN]
knode02	k8s-master	10.130.122.10	[DOWN]
knode03	k8s-master	10.130.122.11	[DOWN]
dgx01	dgx	10.130.122.5	[DOWN]
dgx02	dgx	10.130.122.6	[DOWN]
dgx03	dgx	10.130.122.7	[DOWN]
dgx04	dgx	10.130.122.8	[DOWN]

30. Select Shared Storage from the cmha-setup menu.

In this final HA configuration step, cmha-setup will copy the /cm/shared and /home directories to the shared storage, and it configures both head nodes and all cluster nodes to mount it.



31. Select NAS.



32. Select both /cm/shared and /home.

Please select resources that will be shared:

33. Provide the IP address of the NAS host, and the path that the /cm/shared and /home directories should be copied to on the shared storage.

In this case, /var/nfs/general is exported, so the /cm/shared directory will be copied to 10.130.122.252:/var/nfs/general/cmshared, and it will be mounted over /cm/shared on the cluster nodes.

Please fill in NAS parameters	
NAS nost: Path to /cm/shared: Path to /home:	10.130.122.252 /var/nfs/general/cmshared /var/nfs/general/home
< NEXT >	< BACK >

- 34. The wizard shows a summary of the information that it has collected. Press ENTER to continue.
- 35. Select YES when prompt to proceed with the setup.



36. The <code>cmha-setup</code> wizard proceeds with its work. When it completes, select <code>ENTER</code> to finish HA setup.

Preparing r	nas setup 100% 	

The progress is shown below:

Copying NAS data	[OK]
Mount NAS storage	[OK]
Remove old fsmounts	[OK]
Add new fsmounts	[OK]
Remove old fsexports	[OK]
Write NAS mount/unmount scripts	[OK]
Copy mount/unmount scripts	[OK]
Press any key to continue			

37. cmha-setup is now complete. <EXIT> the wizard to return to the shell prompt.

SUMMARY NAS shared storage config
NAS Host: 10.130.122.252
Path to /cm/shared: /var/nfs/general/cmshared Path to /home: /var/nfs/general/home

3.1.1 Verify HA Setup

1. Run the cmha status command to verify that the failover configuration is correct and working as expected.

Note that the command tests the configuration from both directions: from the primary head node to the secondary, and from the secondary to the primary. The active head node is indicated by an asterisk.

```
# cmha status
Node Status: running in active mode
basepod-head1* -> basepod-head2
mysql [ OK ]
ping [ OK ]
basepod-head2 -> basepod-head1*
mysql [ OK ]
ping [ OK ]
status [ OK ]
```

2. Verify that the /cm/shared and /home directories are being mounted from the NAS server.

mount

```
. . . some output omitted . . .
10.130.122.252:/var/nfs/general/cmshared on /cm/shared type nfs4
(rw,relatime,vers=4.2,rsize=32768,wsize=32768,namlen=255,hard,proto=tcp,timeo=600
,retrans=2,sec=sys,clientaddr=10.130.122.253,local_lock=none,addr=10.130.122.252)
10.130.122.252:/var/nfs/general/home on /home type nfs4
(rw,relatime,vers=4.2,rsize=32768,wsize=32768,namlen=255,hard,proto=tcp,timeo=600
,retrans=2,sec=sys,clientaddr=10.130.122.253,local_lock=none,addr=10.130.122.252)
```

3. Login to the head node to be made active and run cmha makeactive.

ssh basepod-head2
cmha makeactive

```
This is the passive head node. Please confirm that this node should
become
the active head node. After this operation is complete, the HA status of
the head nodes will be as follows:
basepod-head2 will become active head node (current state: passive)
basepod-head1 will become passive head node (current state: active)
Continue(c)/Exit(e)? c
Initiating failover...... [ OK ]
```

basepod-head2 is now active head node, makeactive successful

4. Run the cmha status command again to verify that the secondary head node has become the active head node.

```
basepod-head2* -> basepod-head1
    mysql [ OK ]
    ping [ OK ]
status [ OK ]
  basepod-head1 -> basepod-head2*
    mysql [ OK ]
              [ OK ]
    ping
   status [ OK ]
5. Manually failover back to the primary head node.
  # ssh basepod-head1
  # cmha makeactive
   _____
  This is the passive head node. Please confirm that this node should become
  the active head node. After this operation is complete, the HA status of
  the head nodes will be as follows:
  basepod-head1 will become active head node (current state: passive)
  basepod-head2 will become passive head node (current state: active)
  _____
  Continue(c)/Exit(e)? c
  Initiating failover..... [ OK ]
  basepod-head1 is now active head node, makeactive successful
6. Run cmsh status again to verify that the primary head node has become the active
  head node.
```

cmha status Node Status: running in active mode basepod-head1* -> basepod-head2 mysql [OK] ping [OK] status [OK] basepod-head2 -> basepod-head1* mysql [OK] ping [OK] status [OK]

cmha status

Node Status: running in active mode

7. Power on the cluster nodes.

# cmsł	n −c	"powe	er -c	k8s	s-mas	ste	r,dgx	on"	
ipmi0						[ON]	knode01
ipmi0						[ON]	knode02
ipmi0						[ON]	knode03
ipmi0						[ON]	dgx01
ipmi0						[ON]	dgx02
ipmi0						[ON]	dgx03
ipmi0						[ON]	dgx04

8. (Optionally) Configure Jupyter HA

If Jupyter was deployed on the primary head node before HA was configured, configure the Jupyter service to run on the active head node.

	0				
	% device				
	% use basepod-head1				
	% services				
	% use cm-jupyterhub				
	% show				
	Parameter	Value			
	Revision				
	Service	cm-jupyterhub			
	Run if	ALWAYS			
	Monitored	yes			
	Autostart	yes			
	Timeout	-1			
	Belongs to role	yes			
	Sickness check script				
	Sickness check script timeout	10			
	Sickness check interval	60			
9.	Set the runif parameter to activ	e.			
	% set runif active				
	% commit				
	% show				
	Parameter	Value			
	Revision				
	Service	cm-jupyterhub			
	Run if	ACTIVE			
	Monitored	yes			
	Autostart	yes			
	Timeout	-1			
	Belongs to role	yes			
	Sickness check script				
	Sickness check script timeout	10			
	Sickness check interval	60			
10	Configure the Jupyter service on t	the secondary head node.			

% device

9.

- % use basepod-head2
- % services
- % use cm-jupyterhub
- % set runif active

Chapter 4. Basic User Management

BCM uses its own LDAP service to manage users and groups with a centralized LDAP database server running on the head node, and not by entries in /etc/passwd or /etc/group files. An external LDAP server can be setup for authentication services to replace the existing Bright LDAP service, but it is outside of the scope of this document.

Only the basic user management tasks are outlined in this guide to provide a starting point. Refer to the *Bright Cluster Manager Administrator Manual* for complete options and additional details.

Although user management can be done in both cmsh and Bright View, cmsh is used in this chapter.

4.1 Configuring a User

1. Add a user (userone in this case).

```
# cmsh
% user
% add userone
% set password 7adGnv0!K
% commit
```

2. userone will reset the password after successfully logging in.

userone@basepod-head2:~\$ passwd (current) LDAP Password: New password: Retype new password: passwd: password updated successfully userone@basepod-head2:~\$

3. Use show to view user parameters and values.

[basepod-head2->user[userone]]% show			
Parameter	Value		
Accounts			
Managees			
Name	userone		
Primary group	userone		
Revision			
Secondary groups			
ID	1004		
Common name	userone		
Surname	userone		
Group ID	1004		
Login shell	/bin/bash		
Home directory	/home/ userone		
Password	*****		
email			
Profile			
Create cmjob certificate	no		
Write ssh proxy config	no		
Shadow min	0		
Shadow max	999999		
Shadow warning	7		
Inactive	0		
Last change	2022/10/20		
Expiration date	2037/12/31		
Project manager	<submode></submode>		
Notes	<0B>		

4. Use set to change parameters.

[basepod-head2->user	[userone]]% set		
commonname	expirationdate	id	name
profile	shadowmax	surname	
createcmjobcertificate	groupid	inactive	notes
projectmanager	shadowmin	writesshproxyconfig	
email	homedirectory	loginshell	password
revision	shadowwarning		

4.1.1 Procedures to Remove a User

This block of code will delete a user.

```
# cmsh
% user
% remove userone
% commit
```

Adding the -d option to remove will also delete the home directory.

4.2 Adding a User to K8s

To use K8s services, a user must also be added the K8s cluster.

Add each K8s user with cm-kubernetes-setup.

```
root@basepod-head1:~# cm-kubernetes-setup --add-user userone
Connecting to CMDaemon
Executing 10 stages
- kubernetes
 - docker
## Progress: 0
#### stage: kubernetes: Get Kube Cluster
## Progress: 10
#### stage: kubernetes: Check Permissions User Chart
## Progress: 20
#### stage: kubernetes: Check User
## Progress: 30
#### stage: kubernetes: Check Add User
## Progress: 40
#### stage: kubernetes: Check Namespace Does Not Exist
## Progress: 50
#### stage: kubernetes: Check Cluster Admin Has No Operators
## Progress: 60
#### stage: kubernetes: Deploy user
User userone created successfully!
## Progress: 70
#### stage: kubernetes: List Installed Operators
## Progress: 80
#### stage: kubernetes: Update Operator Permissions
## Progress: 90
#### stage: kubernetes: Log Text
User added successfully!
## Progress: 100
Took:
       00:06 min.
Progress: 100/100
Kubernetes Setup finished!
```

4.3 Removing a User from K8s

To remove a user (userone) from K8s, execute this command:

cm-kubernetes-setup --remove-user userone

The user will no longer be able to use the K8s service.

If an attempt is made, this error message will be shown:

Error from server (Forbidden): nodes is forbidden: User "userone" cannot list resource "nodes" in API group "" at the cluster scope

Appendix A. Site Survey

The tables in this section represent responses to a completed site survey and are used as examples in this deployment guide.

Table 4. General information

Item	Value		
NFS server IP	10.130.122.252		
NFS server export	/var/nfs/general		
Head node drive path	nvme0n1		
Cluster name	BasePOD		
Organization	NVIDIA		
Timezone	US/Los_Angeles		
Nameservers	8.8.8.8		
Search domains	example.com		

Table 5. BCM head node information

Item	Value
Head node 1 name	basepod-head1
Head node 1 and 2 administrator password	ExamplePassword1234!@#\$
Head node 1 BMC IP (ipminet)	10.130.111.66
Head node 1 Ethernet device (externalnet)	enp10
Head node 1 IP (externalnet)	10.130.121.254
Head node 1 Ethernet device (internalnet)	enp10
Head node 1 IP (internalnet)	10.130.122.254
Head node 2 name	basepod-head2
Head node 2 BMC IP (ipminet)	10.130.111.67
Head node 2 Ethernet device (externalnet)	enp10
Head node 2 IP (externalnet)	10.130.121.253
Head node 2 Ethernet device (internalnet)	enp10
Head node 2 IP (internalnet)	10.130.122.253

Table 6. Network information

ltem	Value
K8s node name template	knode##
ipminet base IP	10.130.111.64
ipminet netmask	255.255.255.192
ipminet gateway	10.130.111.65
ipminet switch ASN	
<pre>internalnet switch #1 ASN (for externalnet)</pre>	
internalnet switch #2 ASN (for externalnet)	
externalnet base IP	10.130.121.0
externalnet netmask	255.255.255.0
externalnet gateway	10.130.121.1
Domain	example.com
internalnet base IP	10.130.122.0
internalnet netmask	255.255.255.0
ibnet base IP	10.149.0.0
ibnet netmask	255.255.0.0

Table 7. DGX node information

Item	Value
DGX Node 1 name	dgx01
DGX Node 1 MAC (enp225s0f0/Management)	B8:CE:F6:2F:08:69
DGX Node 1 MAC (enp97s0f0/Management)	B8:CE:F6:2D:0E:A7
DGX Node 2 name	dgx02
DGX Node 2 MAC (enp225s0f0/Management)	B8:CE:F6:2F:08:69
DGX Node 2 MAC (enp97s0f0/Management)	B8:CE:F6:2D:0E:A7
DGX Node 3 name	dgx03
DGX Node 3 MAC (enp225s0f0/Management)	B8:CE:F6:2F:08:69
DGX Node 3 MAC (enp97s0f0/Management)	B8:CE:F6:2D:0E:A7
DGX Node 4 name	dgx04
DGX Node 4 MAC (enp225s0f0/Management)	B8:CE:F6:2F:08:69
DGX Node 4 MAC (enp97s0f0/Management)	B8:CE:F6:2D:0E:A7

Table 8. K8s node information

ltem	Value
K8S Node 1 interface 1 (Management)	enslflnpl
K8S Node 1 interface 2 (Management)	ens2f1np1
K8S Node 1 interface 1 (Management) MAC	04:3F:72:E7:64:97
K8S Node 1 interface 2 (Management) MAC	OC:42:A1:79:9B:15
K8S Node 2 interface 1 (Management)	enslflnpl
K8S Node 2 interface 2 (Management)	ens2f1np1
K8S Node 2 interface 1 (Management) MAC	04:3F:72:E7:64:97
K8S Node 2 interface 2 (Management) MAC	OC:42:A1:79:9B:15
K8S Node 3 interface 1 (Management)	enslflnpl
K8S Node 3 interface 2 (Management)	ens2f1np1
K8S Node 3 interface 1 (Management) MAC	04:3F:72:E7:64:97
K8S Node 3 interface 2 (Management) MAC	0C:42:A1:79:9B:15

Appendix B. Switch Configurations

Switch configuration files are captured in this section.

B.1 SN4600 #1 (In-band Management Switch)

```
# Note Make sure to update the IP addresses in the sample config below
# eth0 mgmt interface configs
nv set interface eth0 ip address 10.130.111.78/24
nv set interface eth0 ip gateway 10.130.111.65
nv set interface eth0 ip vrf mgmt
nv set interface eth0 type eth
# Creating SVI interfaces and adding the VLANs to the bridge
# Note Make sure to update the IP addresses
nv set bridge domain br default vlan 122
nv set bridge domain br default vlan 121
nv set interface vlan121 type svi
nv set interface vlan121 ip vrr mac-address 44:38:39:ff:ff:ff
nv set interface vlan121 ip vrr address 10.130.121.1/24
nv set interface vlan121 ip address 10.130.121.2/24
nv set interface vlan122 type svi
nv set interface vlan122 ip vrr mac-address 44:38:39:ff:ff:ff
nv set interface vlan122 ip vrr address 10.130.122.1/24
nv set interface vlan122 ip address 10.130.122.2/24
# MLAG configs
nv set interface peerlink bond member swp57
nv set interface peerlink bond member swp58
nv set interface peerlink type peerlink
nv set interface peerlink.4094 base-interface peerlink
nv set interface peerlink.4094 type sub
nv set interface peerlink.4094 vlan 4094
nv set mlag backup 10.130.111.79 vrf mgmt
nv set mlag enable on
nv set mlag mac-address 44:38:39:ff:ff:ff
nv set mlag peer-ip linklocal
nv set mlag priority 2048
### bright headnode 01
nv set interface swp1 bridge domain br default access 121
### bright headnode 02
nv set interface swp2 bridge domain br default access 121
### bond4-20 used to connect to DGX and Kubernetes master nodes
nv set interface bond4 bond member swp4
nv set interface bond4 bond mlag id 4
nv set interface bond4 bridge domain br default untagged 122
nv set interface bond4 bridge domain br default vlan all
nv set interface bond4 bond mlag enable on
nv set interface bond4 bond lacp-bypass on
nv set interface bond5 bond member swp5
nv set interface bond5 bond mlag id 5
nv set interface bond5 bridge domain br default untagged 122
```

```
nv set interface bond5 bridge domain br default vlan all
nv set interface bond5 bond mlag enable on
nv set interface bond5 bond lacp-bypass on
nv set interface bond6 bond member swp6
nv set interface bond6 bond mlag id 6
nv set interface bond6 bridge domain br default untagged 122
nv set interface bond6 bridge domain br default vlan all
nv set interface bond6 bond mlag enable on
nv set interface bond6 bond lacp-bypass on
nv set interface bond7 bond member swp7
nv set interface bond7 bond mlag id 7
nv set interface bond7 bridge domain br_default untagged 122
nv set interface bond7 bridge domain br default vlan all
nv set interface bond7 bond mlag enable on
nv set interface bond7 bond lacp-bypass on
nv set interface bond8 bond member swp8
nv set interface bond8 bond mlag id 8
nv set interface bond8 bridge domain br default untagged 122
nv set interface bond8 bridge domain br default vlan all
nv set interface bond8 bond mlag enable on
nv set interface bond8 bond lacp-bypass on
nv set interface bond9 bond member swp9
nv set interface bond9 bond mlag id 9
nv set interface bond9 bridge domain br default untagged 122
nv set interface bond9 bridge domain br default vlan all
nv set interface bond9 bond mlag enable on
nv set interface bond9 bond lacp-bypass on
nv set interface bond10 bond member swp10
nv set interface bond10 bond mlag id 10
nv set interface bond10 bridge domain br default untagged 122
nv set interface bond10 bridge domain br default vlan all
nv set interface bond10 bond mlag enable on
nv set interface bond10 bond lacp-bypass on
nv set interface bond11 bond member swp11
nv set interface bond11 bond mlag id 11
nv set interface bond11 bridge domain br default untagged 122
nv set interface bond11 bridge domain br default vlan all
nv set interface bond11 bond mlag enable on
nv set interface bond11 bond lacp-bypass on
nv set interface bond12 bond member swp12
nv set interface bond12 bond mlag id 12
nv set interface bond12 bridge domain br default untagged 122
nv set interface bond12 bridge domain br default vlan all
nv set interface bond12 bond mlag enable on
nv set interface bond12 bond lacp-bypass on
nv set interface bond13 bond member swp13
nv set interface bond13 bond mlag id 13
nv set interface bond13 bridge domain br default untagged 122
nv set interface bond13 bridge domain br default vlan all
nv set interface bond13 bond mlag enable on
nv set interface bond13 bond lacp-bypass on
nv set interface bond14 bond member swp14
nv set interface bond14 bond mlag id 14
nv set interface bond14 bridge domain br default untagged 122
nv set interface bond14 bridge domain br default vlan all
nv set interface bond14 bond mlag enable on
nv set interface bond14 bond lacp-bypass on
```

```
nv set interface bond15 bond member swp15
nv set interface bond15 bond mlag id 15
nv set interface bond15 bridge domain br default untagged 122
nv set interface bond15 bridge domain br default vlan all
nv set interface bond15 bond mlag enable on
nv set interface bond15 bond lacp-bypass on
nv set interface bond16 bond member swp16
nv set interface bond16 bond mlag id 16
nv set interface bond16 bridge domain br default untagged 122
nv set interface bond16 bridge domain br default vlan all
nv set interface bond16 bond mlag enable on
nv set interface bond16 bond lacp-bypass on
nv set interface bond17 bond member swp17
nv set interface bond17 bond mlag id 17
nv set interface bond17 bridge domain br default untagged 122
nv set interface bond17 bridge domain br default vlan all
nv set interface bond17 bond mlag enable on
nv set interface bond17 bond lacp-bypass on
nv set interface bond18 bond member swp18
nv set interface bond18 bond mlag id 18
nv set interface bond18 bridge domain br default untagged 122
nv set interface bond18 bridge domain br default vlan all
nv set interface bond18 bond mlag enable on
nv set interface bond18 bond lacp-bypass on
nv set interface bond19 bond member swp19
nv set interface bond19 bond mlag id 19
nv set interface bond19 bridge domain br default untagged 122
nv set interface bond19 bridge domain br default vlan all
nv set interface bond19 bond mlag enable on
nv set interface bond19 bond lacp-bypass on
nv set interface bond20 bond member swp20
nv set interface bond20 bond mlag id 20
nv set interface bond20 bridge domain br default untagged 122
nv set interface bond20 bridge domain br default vlan all
nv set interface bond20 bond mlag enable on
nv set interface bond20 bond lacp-bypass on
### BGP unnumbered configuration (NOTE: no IPs need to be configured on the BGP
interfaces, when using BGP unnumbered)
nv set router bgp autonomous-system 420000003
nv set router bgp enable on
nv set router bgp router-id 10.130.111.78
nv set vrf default router bgp address-family ipv4-unicast enable on
nv set vrf default router bqp address-family ipv4-unicast redistribute connected enable
on
nv set vrf default router bgp enable on
nv set vrf default router bgp neighbor swp64 remote-as external
nv set vrf default router bgp neighbor swp64 type unnumbered
nv set vrf default router bgp neighbor swp63 remote-as external
nv set vrf default router bgp neighbor swp63 type unnumbered
nv set vrf default router bgp neighbor swp60 remote-as external
nv set vrf default router bgp neighbor swp60 type unnumbered
nv set vrf default router bgp neighbor peerlink.4094 remote-as internal
nv set vrf default router bgp neighbor peerlink.4094 type unnumbered
```

B.2 SN4600 #2 (In-band Management Switch)

```
# Note Make sure to update the IP addresses in the sample config below
# eth0 mgmt interface configs
nv set interface eth0 ip address 10.130.111.79/24
nv set interface eth0 ip gateway 10.130.111.65
nv set interface eth0 ip vrf mgmt
nv set interface eth0 type eth
# Creating SVI interfaces and adding the VLANs to the bridge
# Note Make sure to update the IP addresses
nv set bridge domain br default vlan 122
nv set bridge domain br default vlan 121
nv set interface vlan121 type svi
nv set interface vlan121 ip vrr mac-address 44:38:39:ff:ff:ff
nv set interface vlan121 ip vrr address 10.130.121.1/24
nv set interface vlan121 ip address 10.130.121.3/24
nv set interface vlan122 type svi
nv set interface vlan122 ip vrr mac-address 44:38:39:ff:ff:ff
nv set interface vlan122 ip vrr address 10.130.122.1/24
nv set interface vlan122 ip address 10.130.122.3/24
# MLAG configs
nv set interface peerlink bond member swp57
nv set interface peerlink bond member swp58
nv set interface peerlink type peerlink
nv set interface peerlink.4094 base-interface peerlink
nv set interface peerlink.4094 type sub
nv set interface peerlink.4094 vlan 4094
nv set mlag backup 10.130.111.78 vrf mgmt
nv set mlag enable on
nv set mlag mac-address 44:38:39:ff:ff:ff
nv set mlag peer-ip linklocal
### bright headnode 01
nv set interface swp1 bridge domain br default access 122
### bright headnode 02
nv set interface swp2 bridge domain br default access 122
### bond4-20 used to connect to DGX and Kubernetes master nodes
nv set interface bond4 bond member swp4
nv set interface bond4 bond mlag id 4
nv set interface bond4 bridge domain br_default untagged 122
nv set interface bond4 bridge domain br default vlan all
nv set interface bond4 bond mlag enable on
nv set interface bond4 bond lacp-bypass on
nv set interface bond5 bond member swp5
nv set interface bond5 bond mlag id 5
nv set interface bond5 bridge domain br default untagged 122
nv set interface bond5 bridge domain br default vlan all
nv set interface bond5 bond mlag enable on
nv set interface bond5 bond lacp-bypass on
nv set interface bond6 bond member swp6
nv set interface bond6 bond mlag id 6
nv set interface bond6 bridge domain br default untagged 122
nv set interface bond6 bridge domain br default vlan all
```

```
nv set interface bond6 bond mlag enable on
nv set interface bond6 bond lacp-bypass on
nv set interface bond7 bond member swp7
nv set interface bond7 bond mlag id 7
nv set interface bond7 bridge domain br default untagged 122
nv set interface bond7 bridge domain br default vlan all
nv set interface bond7 bond mlag enable on
nv set interface bond7 bond lacp-bypass on
nv set interface bond8 bond member swp8
nv set interface bond8 bond mlag id 8
nv set interface bond8 bridge domain br default untagged 122
nv set interface bond8 bridge domain br default vlan all
nv set interface bond8 bond mlag enable on
nv set interface bond8 bond lacp-bypass on
nv set interface bond9 bond member swp9
nv set interface bond9 bond mlag id 9
nv set interface bond9 bridge domain br default untagged 122
nv set interface bond9 bridge domain br default vlan all
nv set interface bond9 bond mlag enable on
nv set interface bond9 bond lacp-bypass on
nv set interface bond10 bond member swp10
nv set interface bond10 bond mlag id 10
nv set interface bond10 bridge domain br default untagged 122
nv set interface bond10 bridge domain br default vlan all
nv set interface bond10 bond mlag enable on
nv set interface bond10 bond lacp-bypass on
nv set interface bond11 bond member swp11
nv set interface bond11 bond mlag id 11
nv set interface bondl1 bridge domain br default untagged 122
nv set interface bondl1 bridge domain br default vlan all
nv set interface bond11 bond mlag enable on
nv set interface bond11 bond lacp-bypass on
nv set interface bond12 bond member swp12
nv set interface bond12 bond mlag id 12
nv set interface bond12 bridge domain br default untagged 122
nv set interface bond12 bridge domain br default vlan all
nv set interface bond12 bond mlag enable on
nv set interface bond12 bond lacp-bypass on
nv set interface bond13 bond member swp13
nv set interface bond13 bond mlag id 13
nv set interface bond13 bridge domain br default untagged 122
nv set interface bond13 bridge domain br default vlan all
nv set interface bond13 bond mlag enable on
nv set interface bond13 bond lacp-bypass on
nv set interface bond14 bond member swp14
nv set interface bond14 bond mlag id 14
nv set interface bond14 bridge domain br default untagged 122
nv set interface bond14 bridge domain br default vlan all
nv set interface bond14 bond mlag enable on
nv set interface bond14 bond lacp-bypass on
nv set interface bond15 bond member swp15
nv set interface bond15 bond mlag id 15
nv set interface bond15 bridge domain br default untagged 122
nv set interface bond15 bridge domain br default vlan all
nv set interface bond15 bond mlag enable on
nv set interface bond15 bond lacp-bypass on
nv set interface bond16 bond member swp16
```

```
nv set interface bond16 bond mlag id 16
nv set interface bond16 bridge domain br default untagged 122
nv set interface bond16 bridge domain br default vlan all
nv set interface bond16 bond mlag enable on
nv set interface bond16 bond lacp-bypass on
nv set interface bond17 bond member swp17
nv set interface bond17 bond mlag id 17
nv set interface bond17 bridge domain br default untagged 122
nv set interface bond17 bridge domain br default vlan all
nv set interface bond17 bond mlag enable on
nv set interface bond17 bond lacp-bypass on
nv set interface bond18 bond member swp18
nv set interface bond18 bond mlag id 18
nv set interface bond18 bridge domain br default untagged 122
nv set interface bond18 bridge domain br default vlan all
nv set interface bond18 bond mlag enable on
nv set interface bond18 bond lacp-bypass on
nv set interface bond19 bond member swp19
nv set interface bond19 bond mlag id 19
nv set interface bond19 bridge domain br default untagged 122
nv set interface bond19 bridge domain br default vlan all
nv set interface bond19 bond mlag enable on
nv set interface bond19 bond lacp-bypass on
nv set interface bond20 bond member swp20
nv set interface bond20 bond mlag id 20
nv set interface bond20 bridge domain br default untagged 122
nv set interface bond20 bridge domain br default vlan all
nv set interface bond20 bond mlag enable on
nv set interface bond20 bond lacp-bypass on
### BGP unnumbered configuration (NOTE: no IPs need to be configured on the BGP
interfaces, when using BGP unnumbered)
nv set router bgp autonomous-system 420000003
nv set router bgp enable on
nv set router bgp router-id 10.130.111.79
nv set vrf default router bgp address-family ipv4-unicast enable on
nv set vrf default router bgp address-family ipv4-unicast redistribute connected enable
on
nv set vrf default router bgp enable on
nv set vrf default router bgp neighbor swp63 remote-as external
nv set vrf default router bgp neighbor swp63 type unnumbered
nv set vrf default router bgp neighbor swp64 remote-as external
nv set vrf default router bgp neighbor swp64 type unnumbered
nv set vrf default router bgp neighbor swp60 remote-as external
nv set vrf default router bgp neighbor swp60 type unnumbered
nv set vrf default router bgp neighbor peerlink.4094 remote-as internal
nv set vrf default router bgp neighbor peerlink.4094 type unnumbered
```

B.3

SN2201 (Out-of-band Management Switch)

#eth0 nv set interface eth0 ip address 10.130.111.77/24 nv set interface eth0 ip gateway 10.130.111.65 nv set interface eth0 ip vrf mgmt nv set interface eth0 type eth # Creating SVI interfaces and adding the VLANs to the bridge # Note Make sure to update the IP addresses nv set interface vlan111 ip address 10.130.111.65/26 nv set bridge domain br default vlan 111 ### BGP configurations nv set router bgp autonomous-system 4200000004 nv set router bgp enable on nv set router bgp router-id 10.130.111.77 nv set vrf default router bgp address-family ipv4-unicast enable on nv set vrf default router bgp address-family ipv4-unicast redistribute connected enable on nv set vrf default router bgp enable on nv set vrf default router bgp neighbor swp51 remote-as external nv set vrf default router bgp neighbor swp51 type unnumbered nv set vrf default router bgp neighbor swp52 remote-as external nv set vrf default router bgp neighbor swp52 type unnumbered

interfaces connected to IPMI interfaces of different servers
nv set interface swp1-40 bridge domain br_default access 111

B.4 Ethernet Network Configuration Verifications

Some of the lines in the output have been truncated for readability.

```
### MLAG verifications:
root@TOR-01:mgmt:~# net show clag
The peer is alive
    Our Priority, ID, and Role: 2048 48:b0:2d:cc:b2:bc primary
   Peer Priority, ID, and Role: 32768 48:b0:2d:ca:93:7b secondary
         Peer Interface and IP: peerlink.4094 fe80::4ab0:2dff:feca:937b (linklocal)
                   Backup IP: 10.130.111.79 vrf mgmt (active)
                   System MAC: 44:38:39:ff:ff:ff
root@TOR-01:mgmt:~#
CLAG Interfaces
Our Interface Peer Interface CLAG Id Conflicts Proto-Down Reason
             ------ -----
     bond10 -
                              10
     bond11 -
                             11
                                       _
                                                             _
     bond12 -
                             12
                                      _
    bond13 bond13
                              13
```

bond14 -14
 bond15
 15

 bond16
 bond16
 16

 bond17
 _ _ _ bond17 -17 _ _ bond18 -18 19 bond19 -_ 20 bond20 bond4 bond4 4 _ _ bond5 -5 _ _ bond6 bond6 6 7 bond7 – _ bond8 -8 _ _ _ bond9 9 ### verifying an access port (below is a access port with VLAN set to 111) network-admin@IPMI-01:mgmt:~\$ net show int swp1 Name MAC Speed MTU Mode UP swp1 68:21:5f:4f:14:81 1G 1500 Access/L2 Alias bcm-bootstrap:eth0 All VLANs on L2 Port _____ 111 Untagged _____ 111 ### verifying a bonded interface in trunk mode (note the native VLAN is set to 122) network-admin@TOR-01:mgmt:~\$ net show int bond9 Name MAC Speed MTU Mode -- ----- ------ ----- -----UP bond9 1c:34:da:29:17:54 100G 9216 802.3ad Bond Details ----- -----Bond Mode: 802.3ad Load Balancing: layer3+4 Minimum Links: 1 LACP Sys Priority: 1 LACP Rate: 1 LACP Bypass: Active All VLANs on L2 Port _____ 1,121,122 Untagged _____ 122 #### BGP verifications
cumulus@TOR-01:mgmt:~\$ net show bgp summary show bgp ipv4 unicast summary _____ BGP router identifier 10.130.111.78, local AS number 4200000003 vrf-id 0 BGP table version 3 RIB entries 5, using 1000 bytes of memory Peers 4, using 91 KiB of memory Neighbor V AS MsgRcvd MsgSent TblVer InQ OutQ Up/Down State/PfxRcd PfxSnt TOR-02(peerlink.4094) 4 4200000003 2705 2706 0 0 0 02:15:00 Spine-02(swp64)44200000022655Spine-01(swp63)44200000012753IPMI-01(swp60)44200000042480 2656 0 0 0 02:12:29 0 0 0 02:17:22 2754 2482 0 0 0 02:03:48 Total number of neighbors 4 cumulus@TOR-02:mgmt:~\$ net show bgp summary show bgp ipv4 unicast summary _____ BGP router identifier 10.130.111.79, local AS number 4200000003 vrf-id 0 BGP table version 3 RIB entries 5, using 1000 bytes of memory Peers 4, using 91 KiB of memory Neighbor V AS MsgRcvd MsgSent TblVer InQ OutQ Up/Down State/PfxRcd PfxSnt 2692 2692 TOR-01(peerlink.4094) 4 420000003 0 0 0 02:14:20 Spine-01(swp63)442000000012692Spine-02(swp64)442000000022641IPMI-01(swp60)442000000042467 2692 0 0 0 0.02:14:21 0 0 0 02:11:48 2642 2469 0 0 0 02:03:07 Total number of neighbors 4 cumulus@IPMI-01:mgmt:~\$ net show bgp summary show bgp ipv4 unicast summary _____ BGP router identifier 10.130.111.77, local AS number 4200000004 vrf-id 0 BGP table version 3 RIB entries 5, using 1000 bytes of memory Peers 2, using 46 KiB of memory Neighbor V AS MsgRcvd MsgSent TblVer InQ OutQ Up/Down State/PfxRcd PfxSnt TOR-01(swp51) 4 420000003 2495 2494 0 0 0 02:04:30 TOR-02(swp52) 4 420000003 2495 2494 0 0 0 02:04:30 Total number of neighbors 2

Notice

This document is provided for information purposes only and shall not be regarded as a warranty of a certain functionality, condition, or quality of a product. NVIDIA Corporation ("NVIDIA") makes no representations or warranties, expressed or implied, as to the accuracy or completeness of the information contained in this document and assumes no responsibility for any errors contained herein. NVIDIA shall have no liability for the consequences or use of such information or for any infringement of patents or other rights of third parties that may result from its use. This document is not a commitment to develop, release, or deliver any Material (defined below), code, or functionality.

NVIDIA reserves the right to make corrections, modifications, enhancements, improvements, and any other changes to this document, at any time without notice.

Customer should obtain the latest relevant information before placing orders and should verify that such information is current and complete.

NVIDIA products are sold subject to the NVIDIA standard terms and conditions of sale supplied at the time of order acknowledgement, unless otherwise agreed in an individual sales agreement signed by authorized representatives of NVIDIA and customer ("Terms of Sale"). NVIDIA hereby expressly objects to applying any customer general terms and conditions with regards to the purchase of the NVIDIA product referenced in this document. No contractual obligations are formed either directly or indirectly by this document.

No license, either expressed or implied, is granted under any NVIDIA patent right, copyright, or other NVIDIA intellectual property right under this document. Information published by NVIDIA regarding third-party products or services does not constitute a license from NVIDIA to use such products or services or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property rights of the third party, or a license from NVIDIA under the patents or other intellectual property rights.

Reproduction of information in this document is permissible only if approved in advance by NVIDIA in writing, reproduced without alteration and in full compliance with all applicable export laws and regulations, and accompanied by all associated conditions, limitations, and notices.

THIS DOCUMENT AND ALL NVIDIA DESIGN SPECIFICATIONS, REFERENCE BOARDS, FILES, DRAWINGS, DIAGNOSTICS, LISTS, AND OTHER DOCUMENTS (TOGETHER AND SEPARATELY, "MATERIALS") ARE BEING PROVIDED "AS IS." NVIDIA MAKES NO WARRANTIES, EXPRESSED, IMPLIED, STATUTORY, OR OTHERWISE WITH RESPECT TO THE MATERIALS, AND EXPRESSLY DISCLAIMS ALL IMPLIED WARRANTIES OF NONINFRINGEMENT, MERCHANTABILITY, AND FITNESS FOR A PARTICULAR PURPOSE. TO THE EXTENT NOT PROHIBITED BY LAW, IN NO EVENT WILL NVIDIA BE LIABLE FOR ANY DAMAGES, INCLUDING WITHOUT LIMITATION ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL, PUNITIVE, OR CONSEQUENTIAL DAMAGES, HOWEVER CAUSED AND REGARDLESS OF THE THEORY OF LIABILITY, ARISING OUT OF ANY USE OF THIS DOCUMENT, EVEN IF NVIDIA HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. Notwithstanding any damages that customer might incur for any reason whatsoever, NVIDIA's aggregate and cumulative liability towards customer for the products described herein shall be limited in accordance with the Terms of Sale for the product.

Trademarks

NVIDIA, the NVIDIA logo, DGX, DGX NVIDIA BasePOD, and NVIDIA Base Command Manager are trademarks and/or registered trademarks of NVIDIA Corporation in the U.S. and other countries. Other company and product names may be trademarks of the respective companies with which they are associated.

Copyright

© 2023 NVIDIA Corporation. All rights reserved.

