



使用经验证的架构 BeeGFS on NetApp with E-Series Storage

NetApp
January 27, 2026

This PDF was generated from <https://docs.netapp.com/zh-cn/beegfs/second-gen/beegfs-solution-overview.html> on January 27, 2026. Always check docs.netapp.com for the latest.

目录

- 使用经验证的架构..... 1
 - 概述和要求..... 1
 - 解决方案概述..... 1
 - 架构概述..... 2
 - 技术要求..... 6
 - 查看解决方案 设计..... 9
 - 设计概述..... 9
 - 硬件配置..... 9
 - 软件配置..... 11
 - 设计验证..... 16
 - 规模估算准则..... 21
 - 性能调整..... 23
 - 高容量组件..... 24
 - 部署解决方案..... 25
 - 部署概述..... 25
 - 了解Ansible清单..... 26
 - 查看最佳实践..... 29
 - 部署硬件..... 32
 - 部署软件..... 35
 - 扩展到五个组件以上..... 71
 - 建议的存储池过度配置百分比..... 71
 - 高容量组件..... 72

使用经验证的架构

概述和要求

解决方案概述

NetApp解决方案 上的BeeGFS将BeeGFS并行文件系统与NetApp EF600存储系统相结合、打造了一个可靠、可扩展且经济高效的基础架构、可满足苛刻的工作负载要求。

NVA计划

NetApp解决方案 上的BeeGFS是NetApp验证架构(NVA)计划的一部分、该计划为客户提供针对特定工作负载和使用情形的参考配置和规模估算指导。NVA解决方案经过全面测试和设计、可最大限度地降低部署风险并加快上市速度。

设计概述

基于NetApp的BeeGFS解决方案是一个可扩展的组件架构、可针对各种要求苛刻的工作负载进行配置。无论是处理多个小文件、管理大量大文件操作、还是混合工作负载、都可以对文件系统进行自定义以满足这些需求。高可用性内置在设计中、采用双层硬件结构、支持在多个硬件层进行独立故障转移、并确保性能稳定一致、即使在部分系统降级期间也是如此。BeeGFS文件系统支持在不同的Linux分发版之间实现高性能和可扩展的环境、并为客户端提供一个易于访问的存储命名空间。要了解更多信息，请访问 ["架构概述"](#)。

用例

以下使用情形适用于NetApp解决方案 上的BeeGFS：

- NVIDIA DGX SuperPOD系统采用采用DGX的A100、H100、H200和B200 GPU。
- 人工智能(AI)、包括机器学习(ML)、深度学习(DL)、大规模自然语言处理(NLP)和自然语言理解(NLP)。有关详细信息，请参见 ["BeeGFS for AI：事实与虚构"](#)。
- 高性能计算(HPC)、包括通过MPI (消息传递接口)和其他分布式计算技术加速的应用程序。有关详细信息，请参见 ["BeeGFS为何超越HPC"](#)。
- 应用程序工作负载的特征为：
 - 读取或写入大于1 GB的文件
 - 多个客户端(10、100和1000)读取或写入同一文件
- 多TB或多PB数据集。
- 需要一个存储命名空间的环境、可以优化大小文件的组合。

优势

在NetApp上使用BeeGFS的主要优势包括：

- 提供经验证的硬件设计、可实现硬件和软件组件的全面集成、以确保可预测的性能和可靠性。
- 使用Ansible进行部署和管理、以实现规模化的精简性和一致性。
- 使用E系列性能分析器和BeeGFS插件提供监控和可观察性。有关详细信息，请参见 ["介绍用于监控NetApp"](#)

E系列解决方案的框架"。

- 高可用性采用共享磁盘架构、可提供数据持久性和可用性。
- 支持使用容器和Kubernetes进行现代工作负载管理和流程编排。有关详细信息，请参见 "[Kubernetes认识BeeGFS：一个适应未来需求的投资故事](#)"。

架构概述

NetApp解决方案 上的BeeGFS包括用于确定支持经验证的工作负载所需的特定设备、布线和配置的架构设计注意事项。

构建块架构

根据存储要求、可以使用不同的方式部署和扩展BeeGFS文件系统。例如、以大量小文件为主要特征的使用情形将受益于额外的元数据性能和容量、而大文件较少的使用情形则可能有利于提高实际文件内容的存储容量和性能。这些多种注意事项会影响并行文件系统部署的不同维度、从而增加了文件系统设计和部署的复杂性。

为了应对这些挑战、NetApp设计了一个标准的构建块架构、用于横向扩展其中的每个方面。通常、BeeGFS组件部署在以下三种配置文件之一中：

- 一个基础组件、包括BeeGFS管理、元数据和存储服务
- BeeGFS元数据加上存储组件
- 仅供BeeGFS存储使用的组件

这三个选项之间的唯一硬件更改是使用较小的驱动器来获取BeeGFS元数据。否则、所有配置更改将通过软件应用。借助Ansible作为部署引擎、为特定组件设置所需的配置文件可以使配置任务变得简单明了。

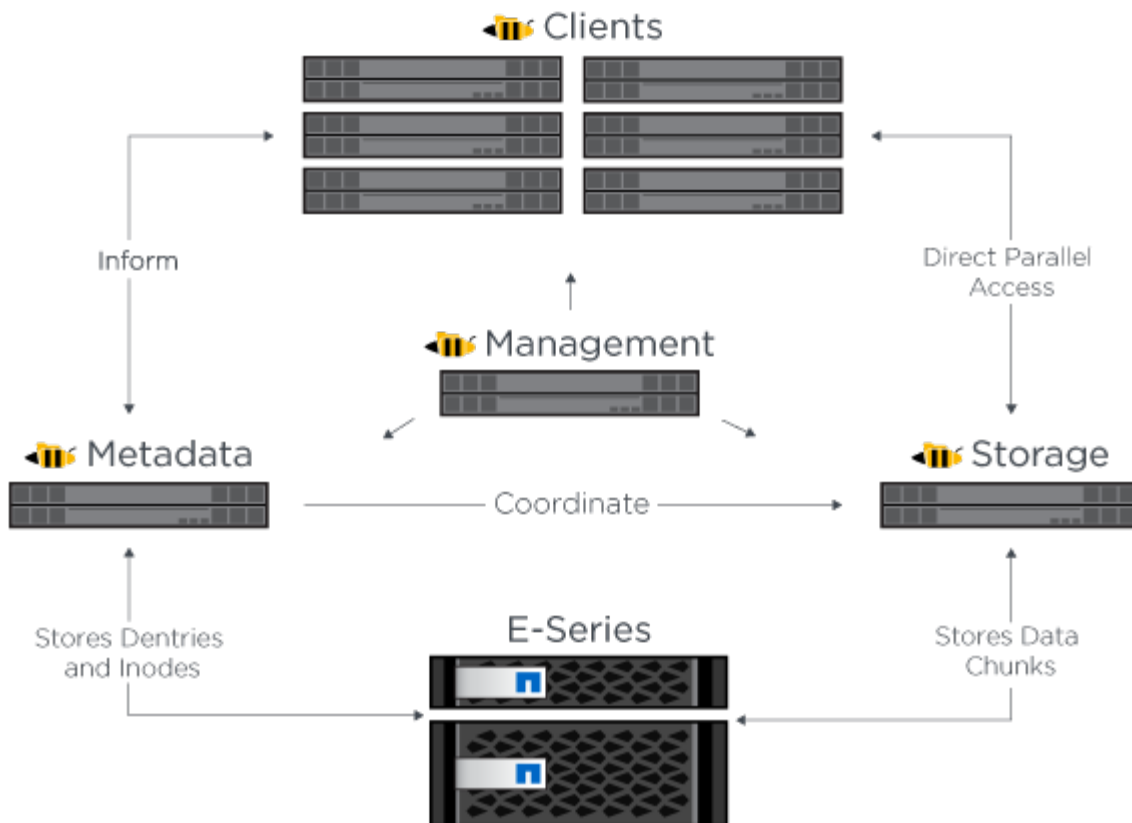
有关更多详细信息、请参见 [\[经验证的硬件设计\]](#)。

文件系统服务

BeeGFS文件系统包括以下主要服务：

- *管理服务。*注册并监控所有其他服务。
- *存储服务。*存储称为数据区块文件的分布式用户文件内容。
- *元数据服务。*跟踪文件系统布局、目录、文件属性等。
- *客户端服务。*挂载文件系统以访问存储的数据。

下图显示了NetApp E系列系统中使用的BeeGFS解决方案 组件和关系。



作为一个并行文件系统、BeeGFS会将其文件条带化到多个服务器节点上、以最大限度地提高读/写性能和可扩展性。服务器节点协同工作、可提供一个文件系统、其他服务器节点(通常称为_clients_)可以同时挂载和访问该文件系统。这些客户端可以查看和使用分布式文件系统、类似于NTFS、XFS或ext4等本地文件系统。

这四项主要服务在广泛支持的Linux分发版上运行、并通过任何支持TCP/IP或RDMA的网络进行通信、包括InfiniBand (IB)、OMNI-Path (OPA)和基于融合以太网的RDMA (RoCE)。BeeGFS服务器服务(管理、存储和元数据)是用户空间守护进程、而客户端是原生 内核模块(无修补)。所有组件均可在不重新启动的情况下安装或更新、您可以在同一节点上运行任何服务组合。

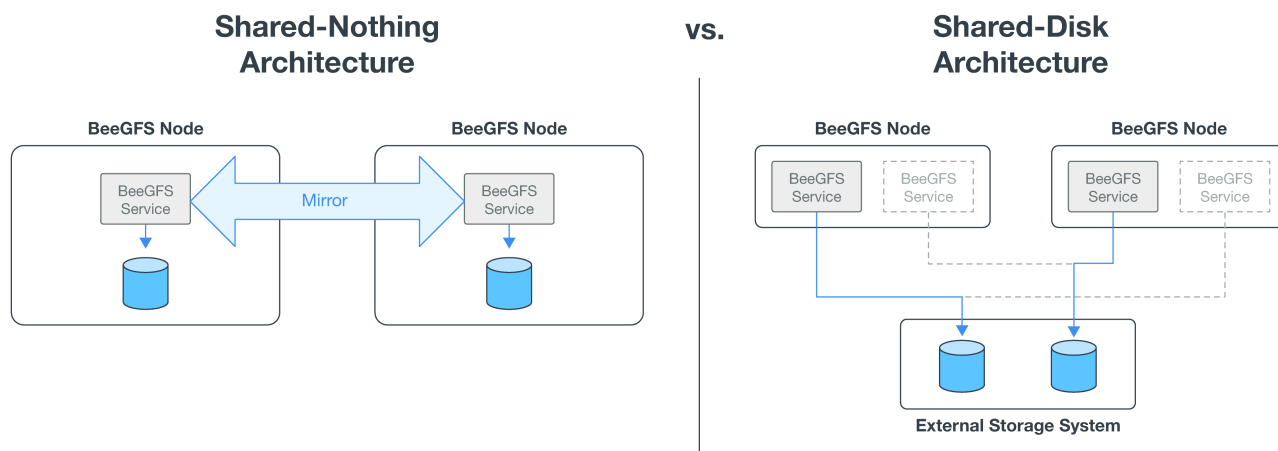
HA架构

NetApp上的BeeGFS通过创建与NetApp硬件完全集成的解决方案 来扩展BeeGFS企业版的功能、从而实现共享磁盘高可用性(HA)架构。



虽然BeeGFS社区版可以免费使用、但企业版要求从NetApp等合作伙伴购买专业支持订阅合同。企业版支持使用多项附加功能、包括故障恢复能力、配额强制实施和存储池。

下图对无共享和共享磁盘HA架构进行了比较。



有关详细信息，请参见 ["宣布NetApp支持的BeeGFS高可用性"](#)。

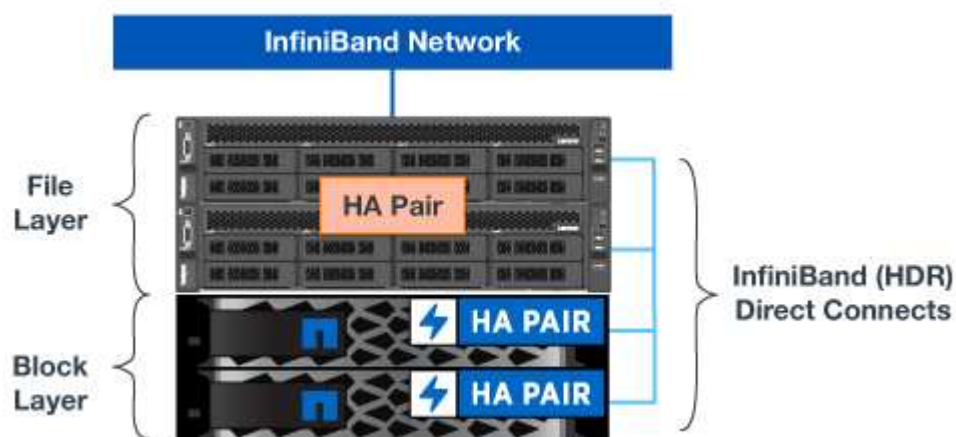
已验证节点

NetApp上的BeeGFS解决方案已验证以下节点。

节点	硬件	详细信息
块	NetApp EF600存储系统	一款高性能全NVMe 2U存储阵列、专为要求苛刻的工作负载而设计。
文件	Lenovo ThinkSystem SR6SR V3服务器	双插槽2U服务器、采用PCIe 5.0、双AMD EPYC 9124处理器。有关Lenovo SR更多功能的详细信息，请参阅 "联想的网站" 。
	Lenovo ThinkSystem SR6SR服务器	双插槽2U服务器、采用PCIe 4.0和双AMD EPYC 7003处理器。有关联想SR想使用的详细信息，请参阅 "联想的网站" 。

经验证的硬件设计

该解决方案的组件(如下图所示)使用经过验证的文件节点服务器作为BeeGFS文件层、并使用两个EF600存储系统作为块层。



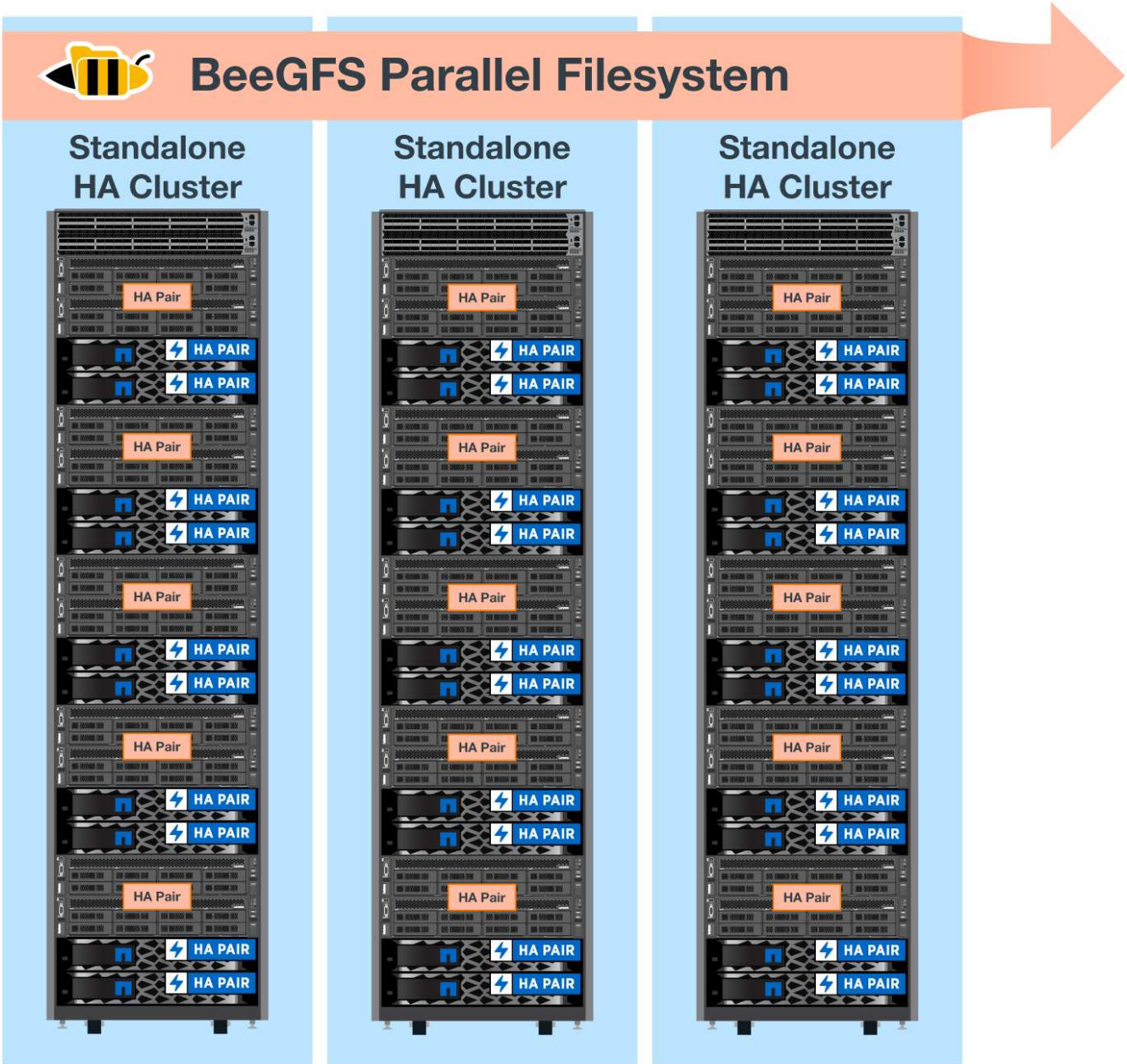
NetApp解决方案上的BeeGFS可跨部署中的所有组件运行。部署的第一个组件必须运行BeeGFS管理、元数据和存储服务(称为基础组件)。所有后续组件均可通过软件进行配置、以扩展元数据和存储服务、或者专门提供存

存储服务。通过这种模块化方法、可以根据工作负载的需求扩展文件系统、同时使用相同的底层硬件平台和组件设计。

最多可以部署五个组件、以构成一个独立的Linux HA集群。这样可以优化PacMaker的资源管理、并保持与Corosync的高效同步。将一个或多个独立BeeGFS HA集群组合在一起、以创建一个BeeGFS文件系统、此文件系统可作为一个存储命名空间供客户端访问。在硬件方面、一个42U机架最多可容纳五个组件、并可为存储/数据网络配备两个1U InfiniBand交换机。有关可视化表示、请参见下图。



要在故障转移集群中建立仲裁、至少需要两个组件。双节点集群存在一些限制、可能会阻止成功进行故障转移。您可以通过将第三个设备整合为Tiebreak 机来配置双节点集群；但是、本文档不会介绍这种设计。



Ansible

NetApp上的BeeGFS是使用Ansible自动化交付和部署的、Ansible自动化托管在GitHub和Ansible GALAXY

上(BeeGFS集合可从获取 "[Ansible Galaxy河](#)" 和 "[NetApp的E系列GitHub](#)") 。虽然Ansible主要是使用用于组装BeeGFS组件的硬件进行测试的、但您可以将其配置为使用受支持的Linux版本在几乎任何基于x86的服务器上运行。

有关详细信息，请参见 "[使用E系列存储部署BeeGFS](#)"。

技术要求

要实施基于NetApp的BeeGFS解决方案、请确保您的环境满足本文档中概述的技术要求。

硬件要求

开始之前、请确保您的硬件满足基于NetApp的BeeGFS解决方案的第二代组件设计的以下规格。根据客户要求、特定部署的确切组件可能会有所不同。

数量	硬件组件	要求
2.	BeeGFS文件节点	每个文件节点都应满足或超出建议文件节点的规格、以实现预期性能。 建议的文件节点选项： • *Lenovo ThinkSystem SR国际 电脑公司 ◦ 处理器： 2个AMD EPYC 9124 16C 3.0 GHz (配置为两个NUMA分区)。 ◦ 内存： 256 GB (16个16 GB TruDR5 4800MHz RDIM-A) ◦ *PCIe扩展： *四个PCIe Gen5 x16插槽(每个NUMA分区两个) ◦ 其他： ▪ 在RAID 1中为操作系统配置两个驱动器(1 TB 7.2K SATA或更高) ▪ 用于带内操作系统管理的1GbE端口 ▪ 采用Redfish API的1GbE BMC、用于带外服务器管理 ▪ 双热插拔电源和性能风扇
2.	E-Series块节点(EF600阵列)	内存： 256 GB (每个控制器128 GB)。适配器： 2端口200GB/HDR (NVMe/IB)。*驱动器： *配置为匹配所需元数据和存储容量。
8.	InfiniBand主机卡适配器(适用于文件节点)。	主机卡适配器可能根据文件节点的服务器型号而有所不同。针对经验证的文件节点的建议包括： • Lenovo ThinkSystem SR6SR V3服务器： ◦ MCX755106AS-热连接X-7、NDR200、QSFP112、双端口、PCIe Gen5 x16、 InfiniBand适配器

数量	硬件组件	要求
1.	存储网络交换机	存储网络交换机必须支持200 Gb/秒的InfiniBand速度。建议的交换机型号包括： * NVIDIA QM9700昆特2 NDNS InfiniBand交换机* * NVIDIA MMM4700多模HDR InfiniBand交换机*

布线要求

从块节点直接连接到文件节点。

数量	部件号	长度
8.	NVIDIA 1500-H001E30 (无源铜缆、QSFP56、200 GB/秒)	1 个月

*从文件节点连接到存储网络交换机。*根据您的InfiniBand存储交换机、从下表中选择适当的缆线选项。+建议的电缆长度为2米；但是、这可能会因客户环境而异。

交换机型号	电缆类型	数量	部件号
NVIDIA QM9700	有源光纤(包括收发器)	2.	MMA4Z00-NS (多模式、IB/ETH、800 Gb/秒2x400 Gb/秒双端口OSFP)
		4	MFP7E20-Nxxx (多模式、4通道到2通道分路器光缆)
		8.	MMA1Z00-NS400 (多模式、IB/ETH、400 Gb/秒单端口QP-112)
	无源铜线	2.	NVIDIA 7Y40-N002 (InfiniBand无源铜缆分路器电缆、InfiniBand 800Gb/s到4个200Gb/s、OSFP到4个QSFP112)
NVIDIA MqM8700	活动光纤	8.	MFS1S00-H003C (NVIDIA有源光纤电缆、InfiniBand 200Gb/秒、QFP56)
	无源铜线	8.	NVIDIA 1500-H002E26 (无源铜缆、InfiniBand 200Gb/秒、QSFP56)

软件和固件要求

为了确保可预测的性能和可靠性、基于NetApp的BeeGFS解决方案版本会使用特定版本的软件和固件组件进行测试。实施该解决方案需要这些版本。

文件节点要求

软件	version
Red Hat Enterprise Linux (RHEL)	RHEL 9.4 高可用性物理服务器（双插槽）。*注：*文件节点需要有效的 Red Hat Enterprise Linux Server 订阅和 Red Hat Enterprise Linux 高可用性附加组件。
Linux内核	5.14.0-427.42.1.el9_4.x86_64

软件	version
HCA固件	ConnectX-7 HCA 固件 FW: 28.45.1200 + PXE: 3.7.0500 + UEFI: 14.38.0016 ConnectX-6 HCA 固件 FW 20.43.2566 + PXE 3.7.0500 + UEFI 14.37.0013

EF600块节点要求

软件	version
SANtricity 操作系统	11.90R3
NVSRAM	N6000-890834-D02.DLP
驱动器固件	适用于所用驱动器型号的最新版本。请参见" E-Series磁盘固件站点 "。

软件部署要求

下表列出了在基于Ansible的BeeGFS部署中自动部署的软件要求。

软件	version
BeeGFS	7.4.6
核心同步	3.1.8-1.
起搏器	2.1.7-5.2.
件	0.11.7-2.
Fence Agent (红鱼/APC)	4.10.0-62.
InfiniBand / RDMA驱动程序	MLNX_OFED_LINUX-23.10-3.2.2.1-LTS

Ansible控制节点要求

NetApp解决方案 上的BeeGFS可从Ansible控制节点进行部署和管理。有关详细信息，请参见 "[Ansible文档](#)"。

下表中列出的软件要求特定于下面列出的NetApp BeeGFS Ansible集合版本。

软件	version
Ansible	10.x
Ansore-core	> 2.13.0
Python	3.10
其他Python软件包	加密法-43.0.0、netaddr-1.3.0、ipaddr-2.2.0
NetApp E-Series BeeGFS折叠资料集	3.2.0

查看解决方案 设计

设计概述

要支持NetApp解决方案 上的BeeGFS、需要使用特定的设备、布线和配置、该BeeGFS并行文件系统与NetApp EF600存储系统相结合。

了解更多信息。

- ["硬件配置"](#)
- ["软件配置"](#)
- ["设计验证"](#)
- ["规模估算准则"](#)
- ["性能调整"](#)

设计和性能各不相同的衍生架构：

- ["高容量组件"](#)

硬件配置

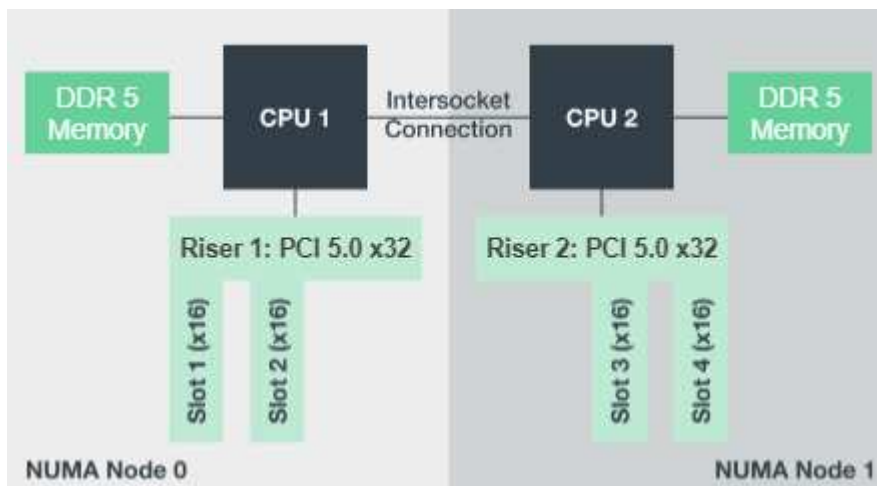
NetApp上BeeGFS的硬件配置包括文件节点和网络布线。

文件节点配置

文件节点具有两个CPU插槽、这些插槽配置为单独的NUMA分区、其中包括对相同数量的PCIe插槽和内存的本地访问。

InfiniBand适配器必须填充到相应的PCI提升板或插槽中、以便在可用的PCIe通道和内存通道上平衡工作负载。您可以通过将单个BeeGFS服务的工作完全隔离到特定NUMA节点来平衡工作负载。目标是从每个文件节点获得类似的性能、就像两个独立的单插槽服务器一样。

下图显示了文件节点NUMA配置。



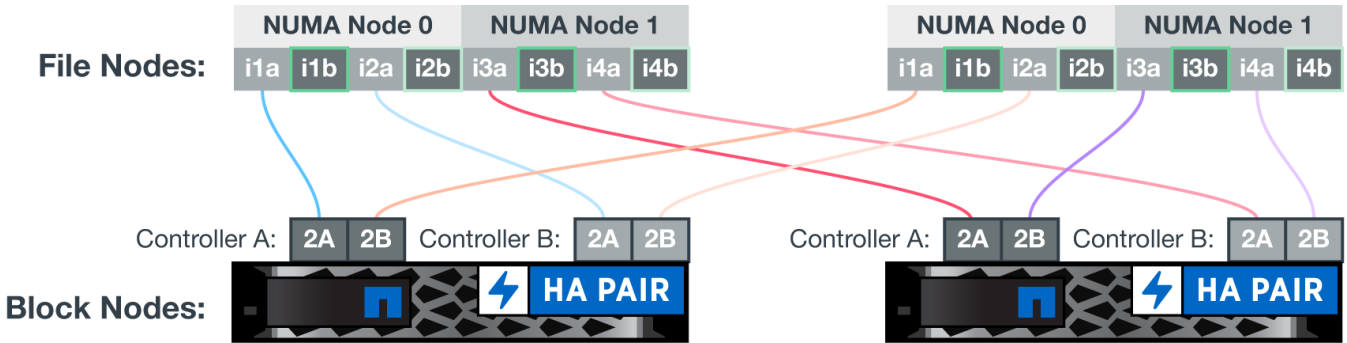
BeeGFS进程固定到特定的NUMA区域、以确保使用的接口位于同一区域。此配置可避免通过插槽间连接进行远程访问。插槽间连接有时称为QPI或GMI2链路；即使在现代处理器架构中、也可能会在使用诸如HDR InfiniBand等高速网络时成为瓶颈。

网络布线配置

在一个构建块中、每个文件节点使用总共四个冗余InfiniBand连接连接到两个块节点。此外、每个文件节点还具有四个到InfiniBand存储网络的冗余连接。

在下图中、请注意：

- 绿色中所显示的所有文件节点端口均用于连接到存储网络结构；所有其他文件节点端口均直接连接到块节点。
- 特定NUMA区域中的两个InfiniBand端口连接到同一块节点的A和B控制器。
- NUMA节点0中的端口始终连接到第一个块节点。
- NUMA节点1中的端口连接到第二个块节点。



在使用分路器缆线将存储交换机连接到文件节点时、应使用一根缆线分支并连接到浅绿色所示的端口。另一根缆线应分支出来并连接到深绿色所示的端口。此外、对于具有冗余交换机的存储网络、浅绿色端口应连接到一个交换机、而深绿色端口应连接到另一个交换机。

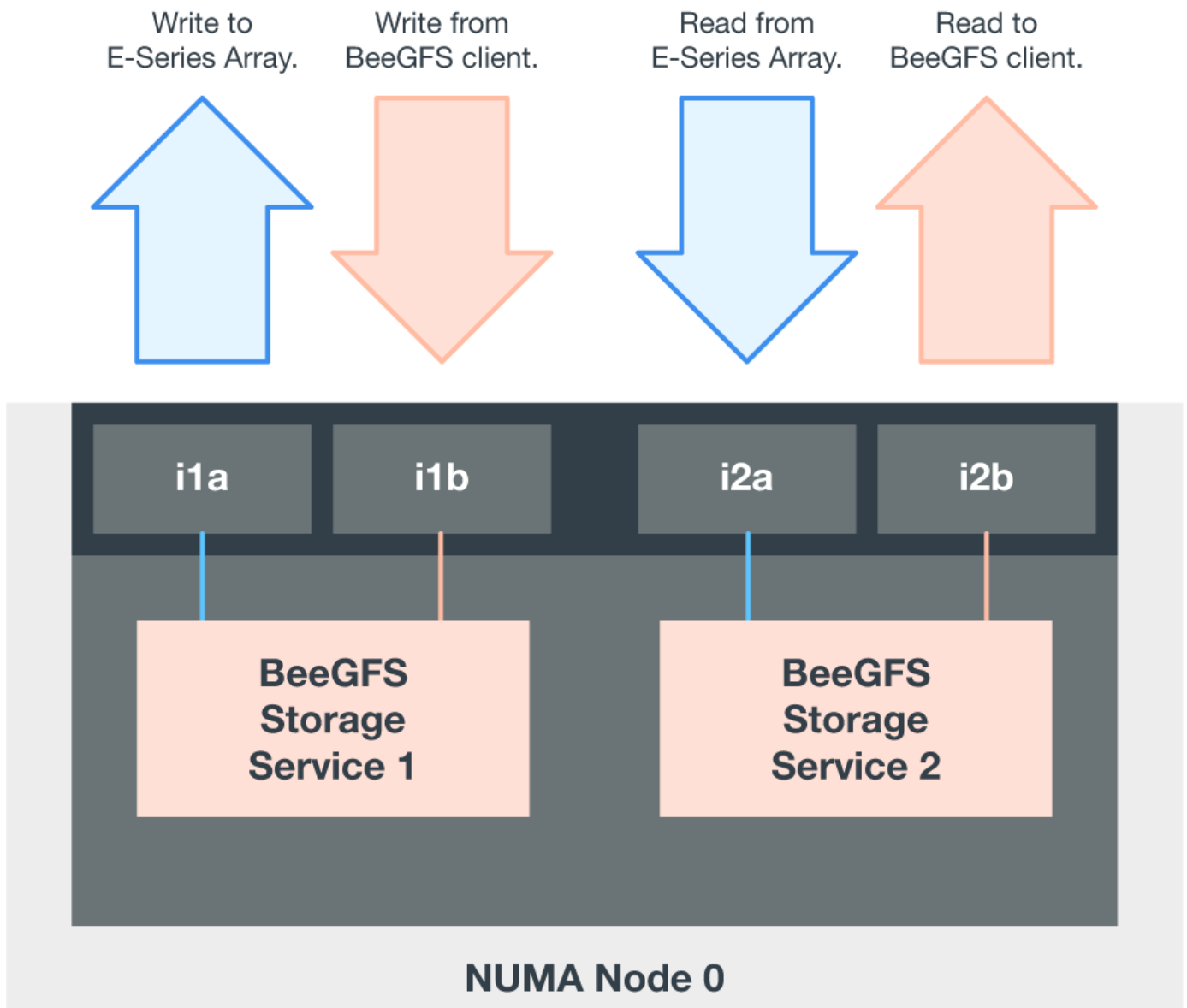
图中所示的布线配置允许每个BeeGFS服务：

- 无论运行BeeGFS服务的文件节点是什么、都可以在同一NUMA区域中运行。
- 无论故障发生在何处、都要有通向前端存储网络和后端块节点的二级最佳路径。
- 如果块节点中的文件节点或控制器需要维护、则最大限度地降低性能影响。

布线以利用带宽

要利用完整的PCIe双向带宽、请确保每个InfiniBand适配器上的一个端口连接到存储网络结构、另一个端口连接到块节点。

下图显示了用于利用完整PCIe双向带宽的布线设计。



对于每个BeeGFS服务、请使用同一适配器将用于客户端流量的首选端口连接到作为该服务卷的主所有者的块节点控制器的路径。有关详细信息，请参见 ["软件配置"](#)。

软件配置

NetApp上的BeeGFS软件配置包括BeeGFS网络组件、EF600块节点、BeeGFS文件节点、资源组和BeeGFS服务。

BeeGFS网络配置

BeeGFS网络配置由以下组件组成。

- ***浮动IP***浮动IP是一种虚拟IP地址、可以动态路由到同一网络中的任何服务器。多个服务器可以拥有相同的浮动IP地址、但在任何给定时间、它只能在一个服务器上处于活动状态。

每个BeeGFS服务器服务都有自己的IP地址、可以根据BeeGFS服务器服务的运行位置在文件节点之间移动。此浮动IP配置允许每个服务单独故障转移到另一个文件节点。客户端只需知道特定BeeGFS服务的IP地

址即可；它不需要知道哪个文件节点当前正在运行该服务。

- * BeeGFS服务器多主机配置*为了提高解决方案 的密度、每个文件节点都有多个存储接口、其IP配置在同一IP子网中。

为了确保此配置在Linux网络堆栈中正常运行、需要进行其他配置、因为默认情况下、如果一个接口的IP位于同一子网中、则可以在其他接口上对对对对请求做出响应。除了其他缺点之外、此默认行为还会使您无法正确建立或维护RDMA连接。

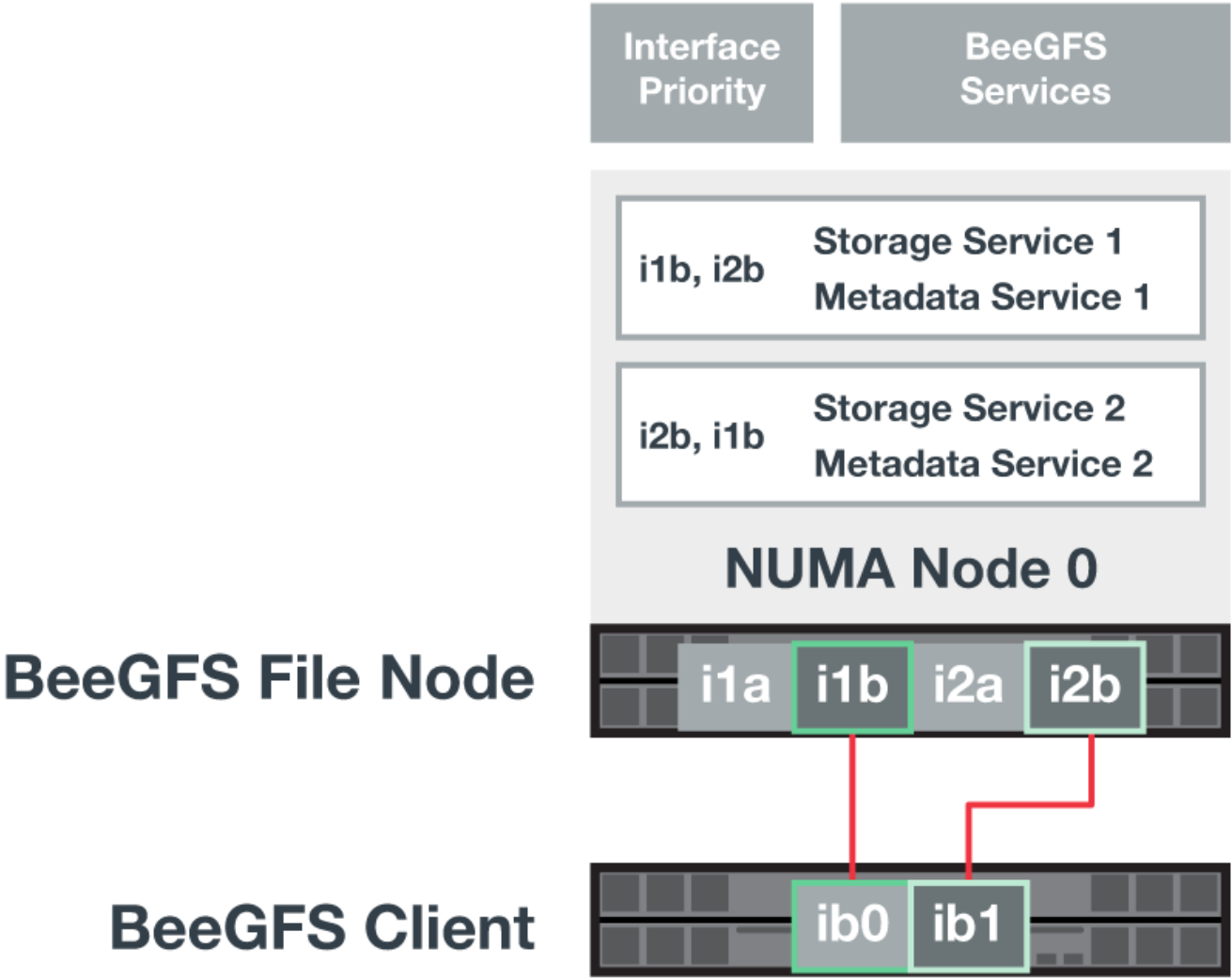
基于Ansible的部署可处理反向路径(RP)和地址解析协议(ARP)行为的收紧、并确保浮动IP何时启动和停止；系统会动态创建相应的IP路由和规则、以使多宿主网络配置正常运行。

- *BeeGFS客户端多轨配置*_Multi-rail_是指应用程序使用多个独立网络连接(或称"导轨")来提高性能的能力。

BeeGFS实施多轨支持、允许在一个IPoIB子网中使用多个IB接口。此功能可实现RDMA NIC之间的动态负载均衡等功能、从而优化网络资源的使用。它还与NVIDIA GPUDirect存储(GDS)相集成、可增加系统带宽、并降低客户端CPU的延迟和利用率。

本文档提供了有关单个IPoIB子网配置的说明。支持双IPoIB子网配置、但其优势与单子网配置不同。

下图显示了在多个BeeGFS客户端接口之间平衡流量的情况。



由于BeeGFS中的每个文件通常在多个存储服务之间进行条带化、因此多导轨配置可以使客户端实现比单个InfiniBand端口更大的吞吐量。例如、以下代码示例显示了一个通用文件条带化配置、该配置允许客户端在两个接口之间平衡流量：

+

```
root@beegfs01:/mnt/beegfs# beegfs-ctl --getentryinfo myfile
Entry type: file
EntryID: 11D-624759A9-65
Metadata node: meta_01_tgt_0101 [ID: 101]
Stripe pattern details:
+ Type: RAID0
+ Chunksize: 1M
+ Number of storage targets: desired: 4; actual: 4
+ Storage targets:
  + 101 @ stor_01_tgt_0101 [ID: 101]
  + 102 @ stor_01_tgt_0101 [ID: 101]
  + 201 @ stor_02_tgt_0201 [ID: 201]
  + 202 @ stor_02_tgt_0201 [ID: 201]
```

EF600块节点配置

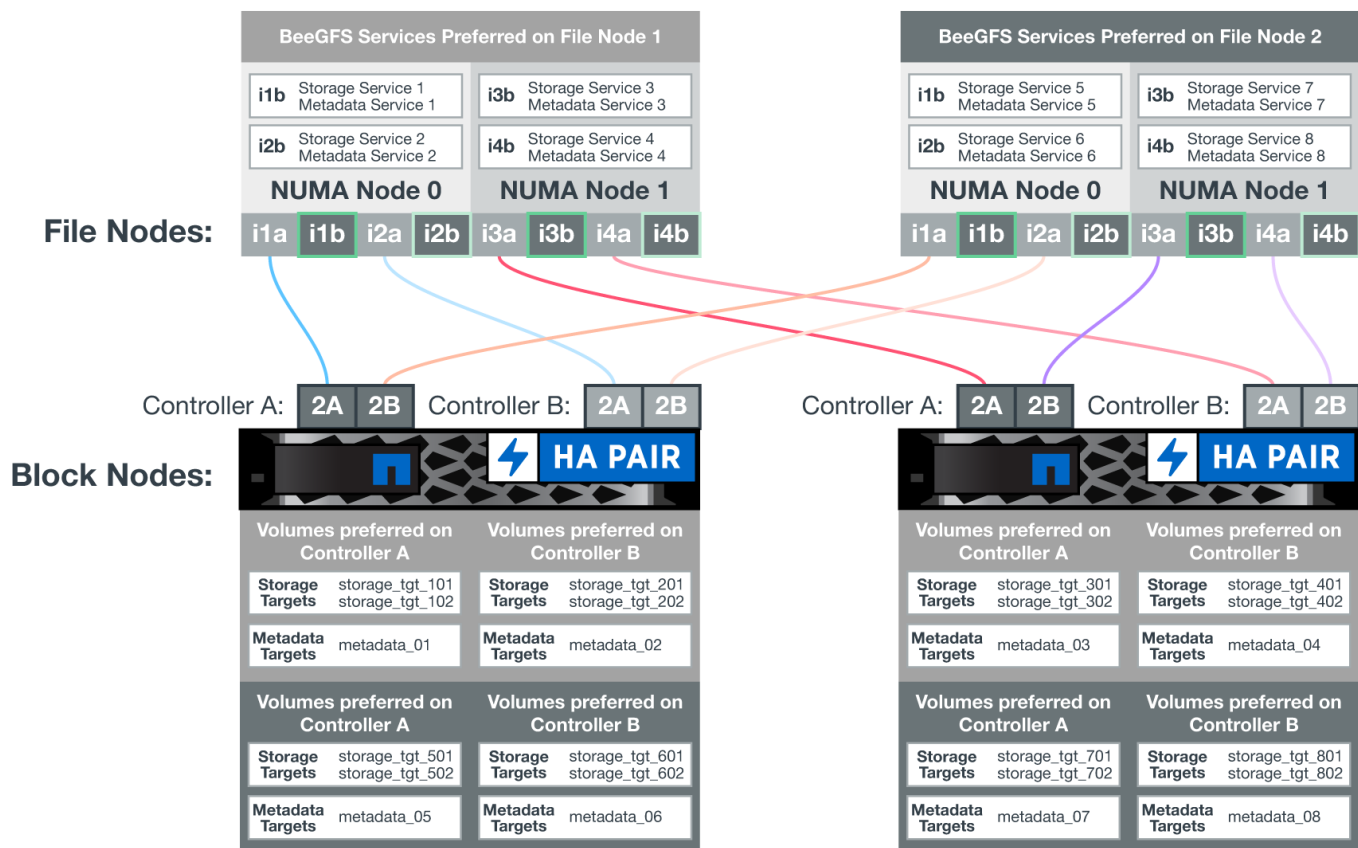
块节点由两个主动/主动RAID控制器组成、这些控制器可共享访问同一组驱动器。通常、每个控制器拥有系统上配置的一半卷、但可以根据需要接管另一个控制器。

文件节点上的多路径软件可确定每个卷的活动和优化路径、并在发生缆线、适配器或控制器故障时自动移至备用路径。

下图显示了EF600块节点中的控制器布局。



为了便于使用共享磁盘HA解决方案、卷会映射到两个文件节点、以便它们可以根据需要相互接管。下图显示了如何配置BeeGFS服务和首选卷所有权以实现最高性能的示例。每个BeeGFS服务左侧的接口指示客户端和其他服务用来与其联系的首选接口。



在上一示例中、客户端和服务更愿使用接口i1b与存储服务1进行通信。存储服务1使用接口i1a作为与其第一个块节点的控制器A上的卷(storage_tgt_101、102)进行通信的首选路径。此配置可利用InfiniBand适配器可用的全双向PCIe带宽、并通过双端口HDR InfiniBand适配器获得比PCIe 4.0更高的性能。

BeeGFS文件节点配置

BeeGFS文件节点配置到高可用性(HA)集群中、以便于在多个文件节点之间对BeeGFS服务进行故障转移。

HA集群设计基于两个广泛使用的Linux HA项目：用于集群成员资格的corosync和用于集群资源管理的Pacemaker。有关详细信息，请参见 ["针对高可用性附加组件的Red Hat培训"](#)。

NetApp编写并扩展了多个开放式集群框架(Open Cluster Framework、OCF)资源代理、使集群能够智能启动和监控BeeGFS资源。

BeeGFS HA集群

通常、在启动BeeGFS服务(无论是否具有HA)时、必须具备一些资源：

- 可访问服务的IP地址、通常由Network Manager配置。
- 用作BeeGFS数据存储目标的底层文件系统。

这些参数通常在`/etc/fstab`中定义、并由systemd挂载。

- 一种系统服务、负责在其他资源准备就绪时启动BeeGFS进程。

如果没有其他软件、这些资源只会在一个文件节点上启动。因此、如果文件节点脱机、则无法访问BeeGFS文件系统的一部分。

由于多个节点可以启动每个BeeGFS服务、因此Pacemaker必须确保每个服务和相关资源一次仅在一个节点上运行。例如、如果两个节点尝试启动同一BeeGFS服务、则在它们都尝试写入底层目标上的相同文件时、存在数据损坏的风险。为了避免这种情况、Pacemaker会依靠Corosync可靠地在所有节点之间保持整个集群的状态同步并建立仲裁。

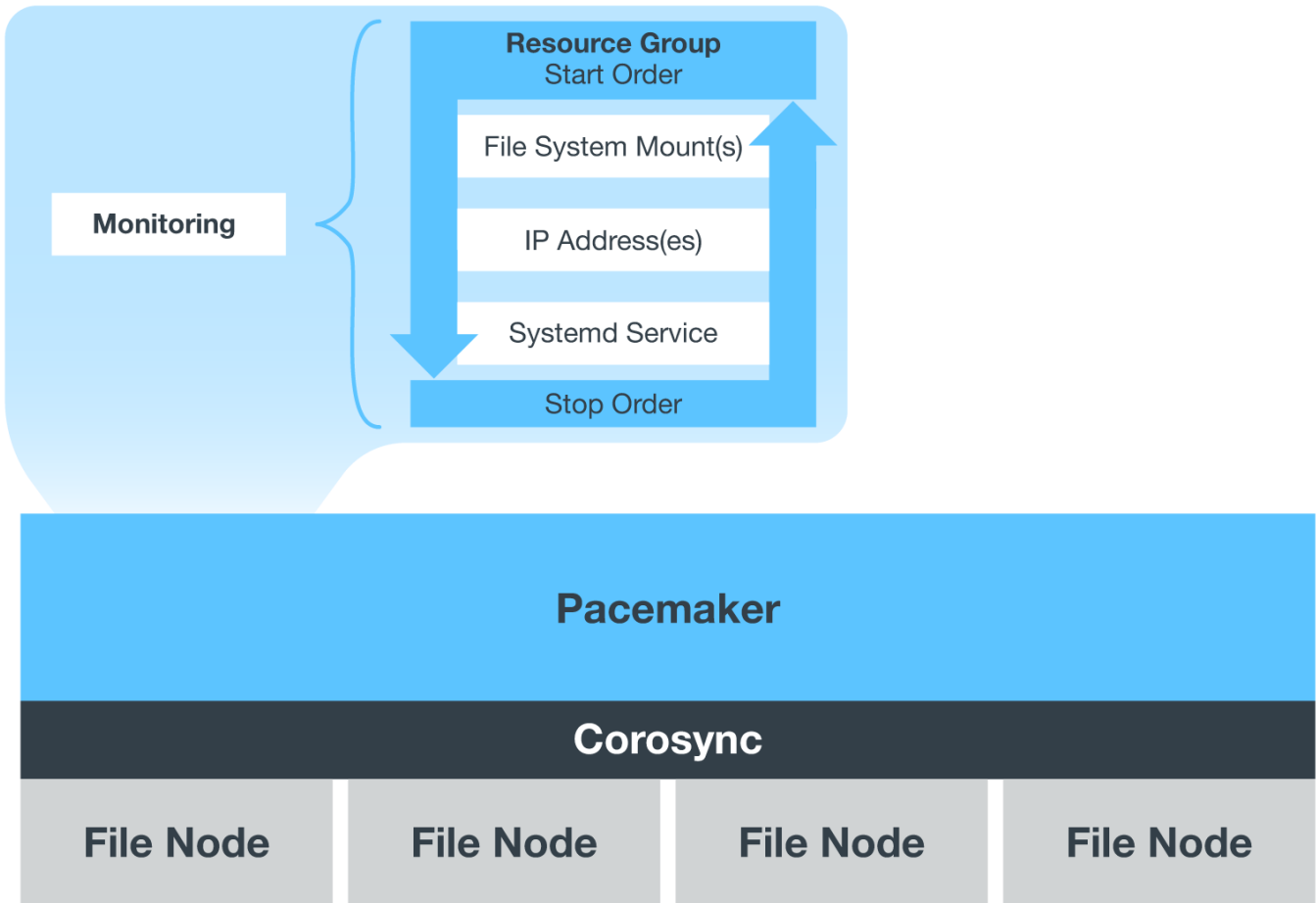
如果集群发生故障、Pacemaker会做出反应、并在另一节点上重新启动BeeGFS资源。在某些情况下、Pacemaker可能无法与原始故障节点通信以确认资源已停止。要在其他位置重新启动BeeGFS资源之前验证节点是否已关闭、Pacemaker最好断开故障节点的电源。

有许多开源隔离代理可用于使Pacemaker使用配电单元(PDU)或使用服务器基板管理控制器(BMC)和Redfo*等API来隔离节点。

当BeeGFS在HA集群中运行时、所有BeeGFS服务和底层资源都由资源组中的Pacemaker管理。每个BeeGFS服务及其所依赖的资源都会配置到一个资源组中、以确保资源按正确顺序启动和停止、并放置在同一节点上。

对于每个BeeGFS资源组、Pacemaker都会运行一个自定义BeeGFS监控资源、该资源负责检测故障情况、并在特定节点上无法再访问BeeGFS服务时智能地触发故障转移。

下图显示了由Pacemaker控制的BeeGFS服务和依赖项。



为了在同一节点上启动多个相同类型的BeeGFS服务、我们将Pacemaker配置为使用多模式配置方法启动BeeGFS服务。有关详细信息，请参见 ["有关多模式的BeeGFS文档"](#)。

由于BeeGFS服务必须能够在多个节点上启动、因此每个服务(通常位于`/etc/beegfs`)的配置文件存储在用作该服

务的BeeGFS目标的E系列卷之一上。这样、可能需要运行此服务的所有节点都可以访问此配置以及特定BeeGFS服务的数据。

```
# tree stor_01_tgt_0101/ -L 2
stor_01_tgt_0101/
├── data
│   ├── benchmark
│   ├── buddymir
│   ├── chunks
│   ├── format.conf
│   ├── lock.pid
│   ├── nodeID
│   ├── nodeNumID
│   ├── originalNodeID
│   ├── targetID
│   └── targetNumID
└── storage_config
    ├── beegfs-storage.conf
    ├── connInterfacesFile.conf
    └── connNetFilterFile.conf
```

设计验证

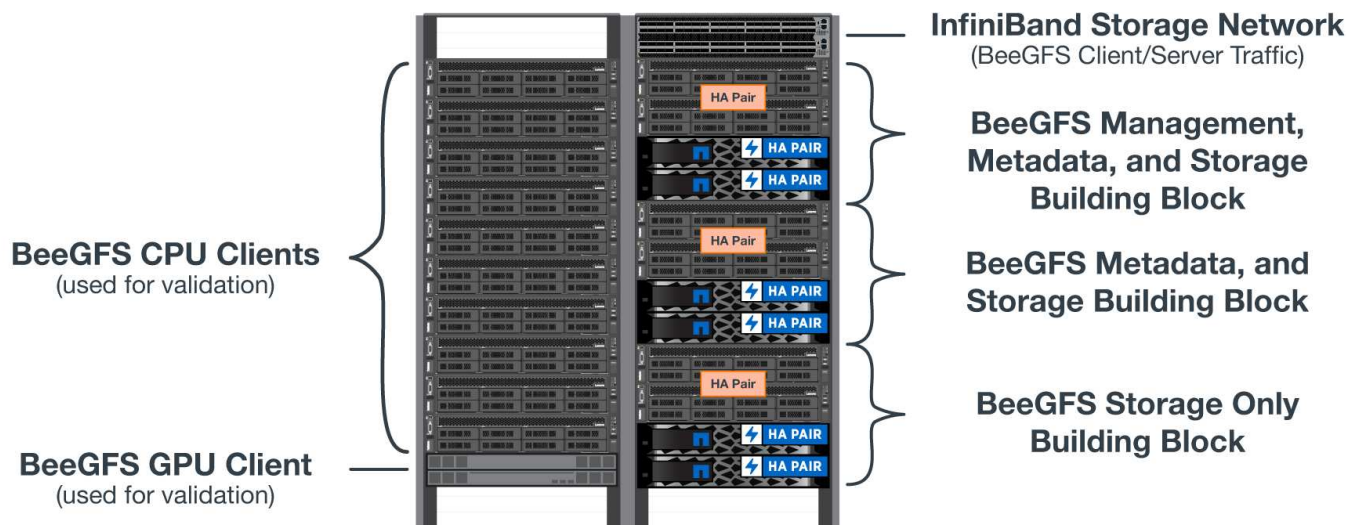
NetApp解决方案 上的BeeGFS的第二代设计已使用三个组件配置文件进行了验证。

配置配置文件包括以下内容：

- 一个基础组件、包括BeeGFS管理、元数据和存储服务。
- BeeGFS元数据加上存储构建块。
- 一个BeeGFS纯存储组件。

这些组件连接到两个NVIDIA qMTM InfiniBand (M8700)交换机。此外、还将10个BeeGFS客户端连接到InfiniBand交换机、并用于运行综合基准实用程序。

下图显示了用于验证NetApp解决方案 上的BeeGFS的BeeGFS配置。



BeeGFS文件条带化

并行文件系统的一个优势是能够在多个存储目标之间对各个文件进行条带化、这可能表示同一个或不同底层存储系统上的卷。

在BeeGFS中、您可以按目录和文件配置条带化、以控制每个文件使用的目标数量、并控制每个文件条带使用的机克大小(或块大小)。此配置允许文件系统支持不同类型的工作负载和I/O配置文件、而无需重新配置或重新启动服务。您可以使用`beegfs-ctl`命令行工具或对使用条带化API的应用程序应用条带设置。有关详细信息、请参见BeeGFS文档 ["条带化"](#) 和 ["条带化API"](#)。

为了获得最佳性能、我们在整个测试过程中对条带模式进行了调整、并记录了每个测试使用的参数。

IOR带宽测试：多个客户端

IOR带宽测试使用OpenMPI运行合成I/O生成器工具IOR的并行作业(可从获取 ["HPC GitHub"](#))连接到一个或多个BeeGFS组件。除非另有说明：

- 所有测试都使用传输大小为1 MiB的直接I/O。
- BeeGFS文件条带化设置为1 MB的机克大小、每个文件一个目标。

以下参数用于IOR、并对区块数进行了调整、以使一个组件的聚合文件大小保持在5 TiB、而三个组件的聚合文件大小保持在40 TiB。

```
mpirun --allow-run-as-root --mca btl tcp -np 48 -map-by node -hostfile
10xnodes ior -b 1024k --posix.odirect -e -t 1024k -s 54613 -z -C -F -E -k
```

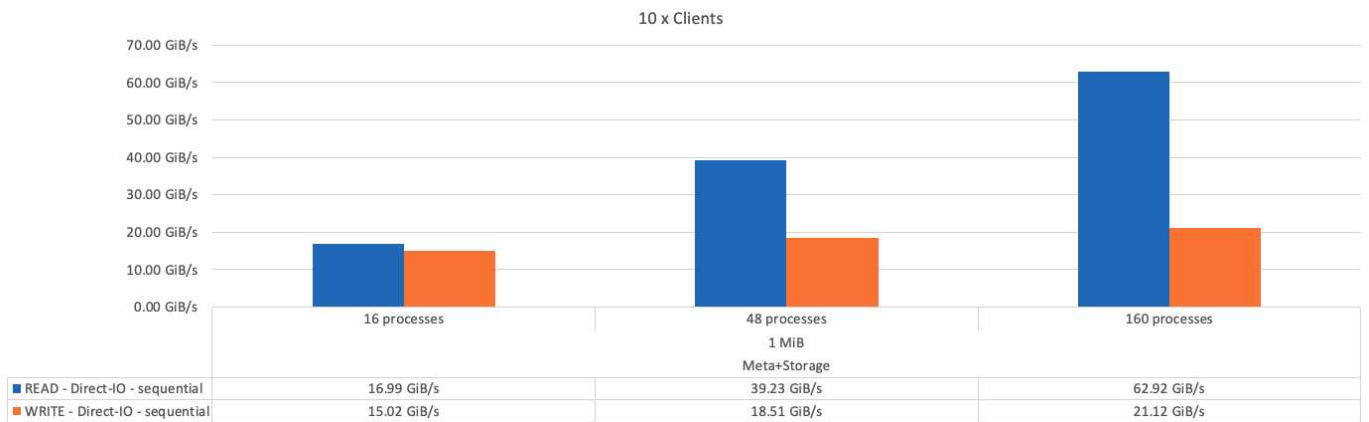
一个BeeGFS基础(管理、元数据和存储)组件

下图显示了一个BeeGFS基础(管理、元数据和存储)组件的IOR测试结果。



BeeGFS元数据+存储组件

下图显示了一个BeeGFS元数据+存储组件的IOR测试结果。



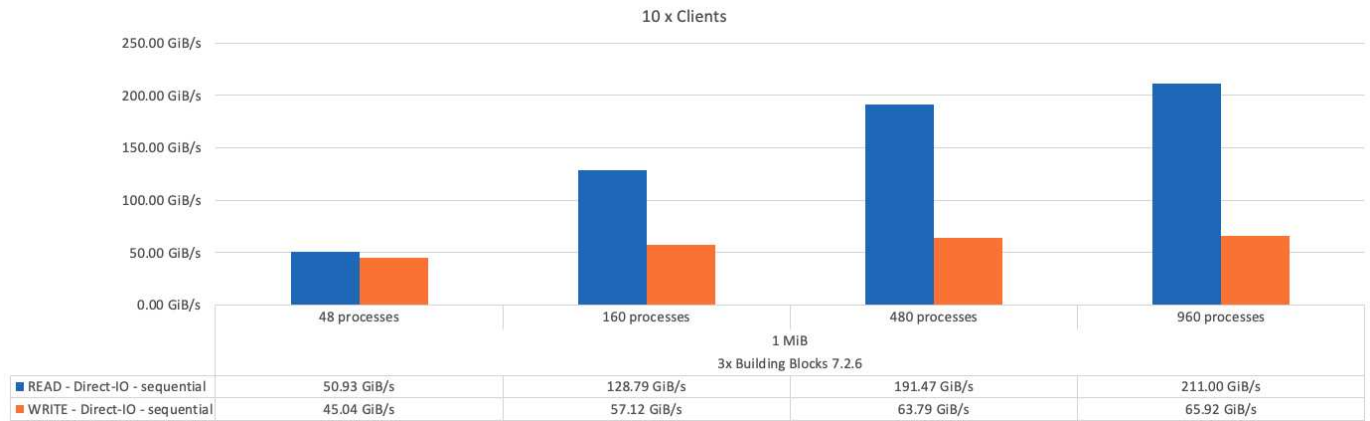
BeeGFS纯存储组件

下图显示了一个BeeGFS纯存储组件的IOR测试结果。



三个BeeGFS组件

下图显示了三个BeeGFS组件的IOR测试结果。



正如预期、基础组件与后续元数据+存储组件之间的性能差异可忽略不计。比较元数据+存储构建块和纯存储构建块后、由于用作存储目标的驱动器增加、读取性能略有提高。但是、写入性能没有显著差异。为了提高性能、您可以将多个组件添加到一起、以线性方式扩展性能。

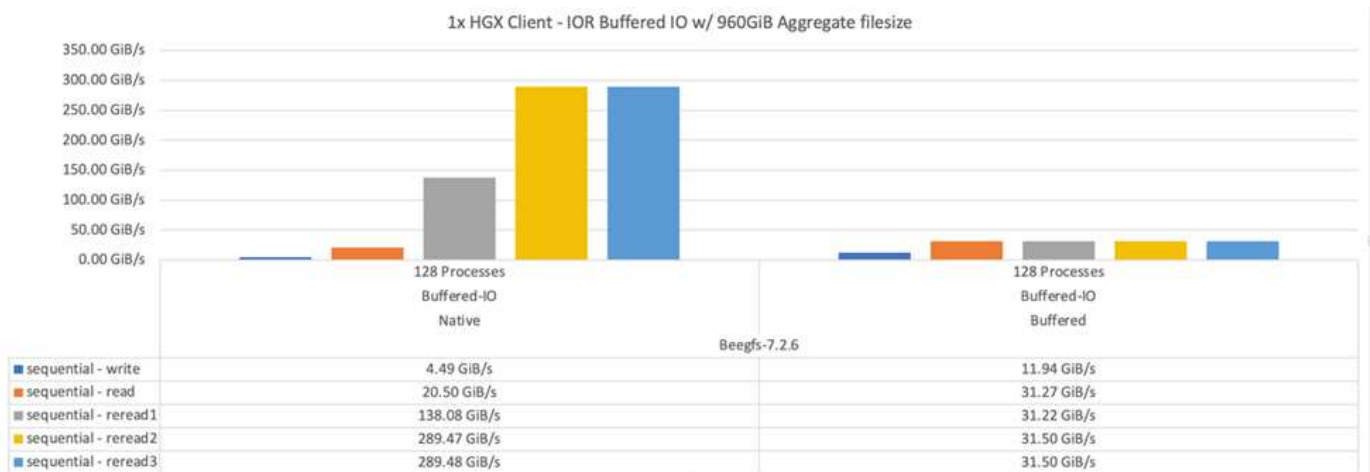
IOR带宽测试：单个客户端

IOR带宽测试使用OpenMPI通过一个高性能GPU服务器运行多个IOR进程、以探索单个客户端可实现的性能。

此测试还会将客户端配置为使用Linux内核页面缓存(tuneFileCacheType =原生)时BeeGFS的重新读取行为和性能与默认的`缓冲区`设置进行比较。

原生 缓存模式会在客户端上使用Linux内核页面缓存、从而使重新读取操作可以来自本地内存、而不是通过网络重新传输。

下图显示了三个BeeGFS组件和一个客户端的IOR测试结果。



在这些测试中、BeeGFS条带化设置为1 MB的机克大小、每个文件有8个目标。

尽管使用默认缓冲模式时写入和初始读取性能较高、但对于多次重新读取相同数据的工作负载、原生 缓存模式会显著提升性能。对于深度学习等工作负载来说、这种提高的重新读取性能非常重要、因为深度学习会在许多时间内多次重新读取同一数据集。

元数据性能测试

元数据性能测试使用MDTest工具(包含在IOR中)测量BeeGFS的元数据性能。这些测试利用OpenMPI在所有十个客户端节点上运行并行作业。

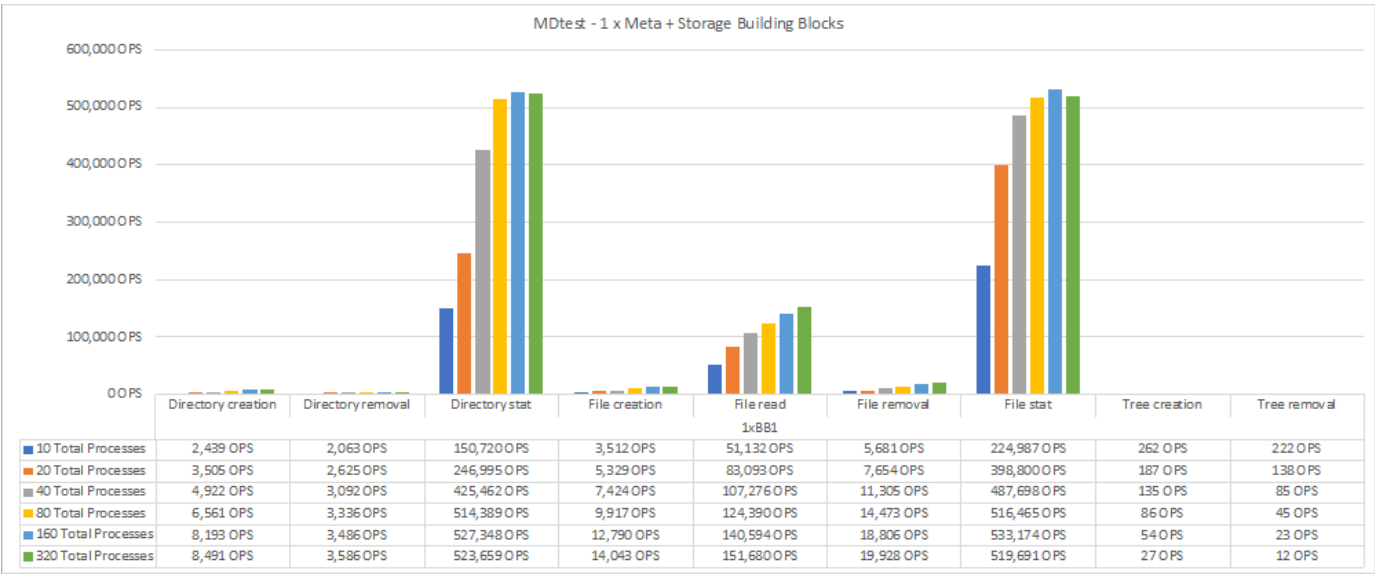
以下参数用于运行基准测试、其中进程总数从10扩展到320、步长为2倍、文件大小为4 k。

```
mpirun -h 10xnodes -map-by node np $processes mdtest -e 4k -w 4k -i 3 -I 16 -z 3 -b 8 -u
```

元数据性能首先使用一到两个元数据+存储组件进行衡量、以显示性能如何通过添加更多组件进行扩展。

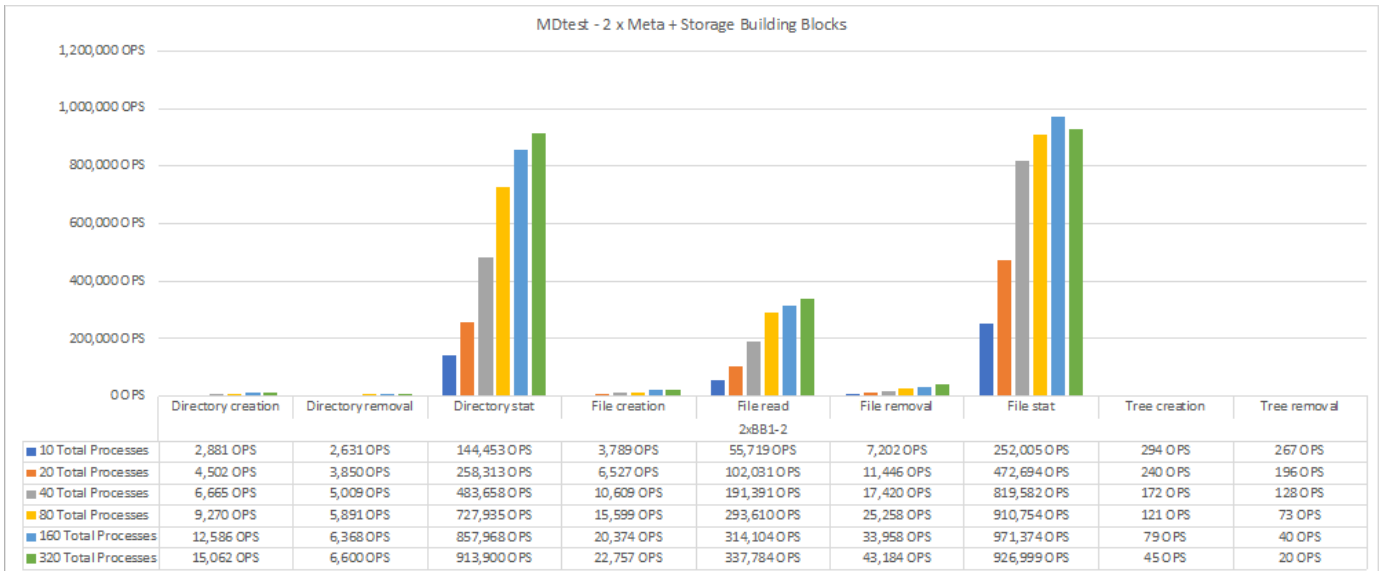
一个BeeGFS元数据+存储组件

下图显示了一个BeeGFS元数据+存储组件的MDTest结果。



两个BeeGFS元数据+存储组件

下图显示了包含两个BeeGFS元数据+存储组件的MDTest结果。



功能验证

在验证此架构的过程中、NetApp执行了多项功能测试、其中包括：

- 通过禁用交换机端口使单个客户端InfiniBand端口出现故障。
- 通过禁用交换机端口使单个服务器InfiniBand端口出现故障。
- 使用BMC触发服务器立即关闭。
- 妥善地将节点置于备用状态并将服务故障转移到另一节点。
- 妥善地将节点重新联机、并将服务故障恢复到原始节点。
- 使用PDU关闭其中一个InfiniBand交换机。所有测试均在压力测试期间执行、并在BeeGFS客户端上设置了`ssysSessionChecksEnabled: false`参数。未发现I/O错误或中断。



存在一个已知的问题描述 (请参见 "[ChangeLog](#)")当BeeGFS客户端/服务器RDMA连接因主接口丢失(如`connInterfacesFile`中所定义)或BeeGFS服务器发生故障而意外中断时、活动客户端I/O可能会挂起多达十分钟、然后才能恢复。如果BeeGFS节点已妥善置于待机状态和待机状态并处于待机状态以进行计划内维护、或者正在使用TCP、则不会发生此问题描述。

NVIDIA DGX SuperPOD和BasePOD验证

NetApp使用一个类似的BeeGFS文件系统验证了用于NVIDIA DGX A100 SuperPOD的存储解决方案、该文件系统由三个组件组成、并应用了元数据和存储配置文件。资格认定工作涉及使用20台DGX A100 GPU服务器测试此NVA所述的解决方案、这些服务器运行各种存储、机器学习和深度学习基准。基于NVIDIA DGX A100 SuperPOD的验证、基于NetApp的BeeGFS解决方案已获得批准、可用于DGX SuperPOD H100、8200和B200系统。此扩展基于满足之前建立的基准和系统要求、并已通过NVIDIA DGX A100的验证。

有关详细信息，请参见 "[采用NetApp技术的NVIDIA DGX SuperPOD](#)" 和 "[NVIDIA DGX基本POD](#)"。

规模估算准则

BeeGFS解决方案 提供了基于验证测试的性能和容量规模估算建议。

构建块架构的目标是、通过添加多个构建块来满足特定BeeGFS系统的要求、创建一个易于调整大小的解决方案

案。您可以根据以下准则估算满足环境要求所需的BeeGFS组件的数量和类型。

请记住、这些估计是最佳性能。我们编写并利用了综合基准测试应用程序、以便以实际应用程序可能无法实现的方式优化底层文件系统的使用。

性能规模估算

下表提供了建议的性能规模估算。

配置配置文件	1 MiB读取	1 MiB写入
元数据+存储	62 GiBps	21 GiBps
仅存储	64 GiBps	21 GiBps

元数据容量估算是根据"经验法则"估算的、即500 GB的容量足以容纳BeeGFS中大约1.5亿个文件。(有关详细信息、请参见BeeGFS文档 ["系统要求"](#))

使用访问控制列表等功能以及每个目录的目录和文件数量也会影响元数据空间的使用速度。存储容量估计值会考虑可用驱动器容量以及RAID 6和XFS开销。

元数据+存储组件的容量规模估算

下表提供了元数据和存储组件的建议容量大小调整。

驱动器大小(2+2 RAID 1)元数据卷组	元数据容量(文件数)	驱动器大小(8+2 RAID 6)存储卷组	存储容量(文件内容)
1.92 TB	1、938、572、200	1.92 TB	51.77 TB
3.84 TB	3、880、388、400	3.84 TB	103.55 TB
7.68 TB	8、125、278、000	7.68 TB	216.74 TB
15.3 TB	17、262、854、000	15.3 TB	460.60 TB



在估算元数据和存储组件的规模时、您可以使用较小的驱动器作为元数据卷组、而不是存储卷组、从而降低成本。

仅存储组件的容量规模估算

下表提供了仅存储组件的容量规模估算规则。

驱动器大小(10+2 RAID 6)存储卷组	存储容量(文件内容)
1.92 TB	59.89 TB
3.84 TB	119.80 TB
7.68 TB	251.89 TB
15.3 TB	538.55 TB



除非启用全局文件锁定、否则将管理服务包含在基本(第一个)组件中所带来的性能和容量开销极小。

性能调整

BeeGFS解决方案 提供了基于验证测试的性能调整建议。

虽然BeeGFS可提供合理的即装即用性能、但NetApp已开发出一组建议的调整参数来最大限度地提高性能。这些参数考虑了底层E系列块节点的功能以及在共享磁盘HA架构中运行BeeGFS所需的任何特殊要求。

文件节点的性能调整

您可以配置的可用调整参数包括：

1. *文件节点的UEFI/BIOS中的系统设置。*为最大限度地提高性能、建议在用作文件节点的服务器型号上配置系统设置。在设置文件节点时、您可以使用由基板管理控制器(BMC)提供的系统设置(UEFI/BIOS)或Redfish API来配置系统设置。

系统设置因用作文件节点的服务器型号而异。必须根据所使用的服务器型号手动配置这些设置。要了解如何配置已验证的 Lenovo SR665 V3 文件节点的系统设置，请参阅["调整文件节点系统设置以提高性能"](#)。

2. *所需配置参数的默认设置。*所需配置参数会影响BeeGFS服务的配置方式以及Pacemaker对E系列卷(块设备)进行格式化和挂载的方式。这些必需的配置参数包括：

- BeeGFS服务配置参数

您可以根据需要覆盖配置参数的默认设置。有关可针对特定工作负载或用例调整的参数，请参见["BeeGFS服务配置参数"](#)。

- 卷格式化和挂载参数设置为建议的默认值、只能针对高级使用情形进行调整。默认值将执行以下操作：

- 根据目标类型(例如管理、元数据或存储)以及底层卷的RAID配置和分段大小优化初始卷格式化。
- 调整Pacemaker挂载每个卷的方式、以确保所做的更改立即刷新到E系列块节点。这样可以防止在文件节点发生故障且正在进行活动写入时丢失数据。

有关可针对特定工作负载或用例调整的参数，请参见 ["卷格式化和挂载配置参数"](#)。

3. *安装在文件节点上的Linux操作系统中的系统设置。*您可以在的步骤4中创建Ands恢复 清单时覆盖默认的Linux操作系统系统设置 ["创建Ansible清单"](#)。

默认设置用于验证NetApp解决方案 上的BeeGFS、但您可以对其进行更改、以便根据您的特定工作负载或使用情形进行调整。您可以更改的Linux操作系统设置示例包括：

- E系列块设备上的I/O队列。

您可以在用作BeeGFS目标的E系列块设备上配置I/O队列、以便：

- 根据设备类型(NVMe、HDD等)调整计划算法。
- 增加未处理的请求数。
- 调整请求大小。
- 优化预读行为。

- 虚拟内存设置。

您可以调整虚拟内存设置、以获得最佳的持续流性能。

- CPU设置。

您可以调整CPU频率调速器和其他CPU配置、以获得最高性能。

- 读取请求大小。

您可以增加NVIDIA HCA的最大读取请求大小。

对块节点进行性能调整

根据应用于特定BeeGFS组件的配置文件、在块节点上配置的卷组会略有变化。例如、对于一个24驱动器EF600块节点：

- 对于单个基础组件、包括BeeGFS管理、元数据和存储服务：
 - 1个2+2 RAID 10卷组、用于BeeGFS管理和元数据服务
 - 2个8+2 RAID 6卷组、用于BeeGFS存储服务
- 对于BeeGFS元数据+存储构建块：
 - 1个2+2 RAID 10卷组、用于BeeGFS元数据服务
 - 2个8+2 RAID 6卷组、用于BeeGFS存储服务
- 仅限BeeGFS存储组件：
 - 2个10+2 RAID 6卷组、用于BeeGFS存储服务



与存储相比、BeeGFS对管理和元数据所需的存储空间要少得多、因此可以选择为RAID 10卷组使用较小的驱动器。较小的驱动器应填充到最外侧的驱动器插槽中。有关详细信息，请参见 ["部署说明"](#)。

所有这些设置均由基于Ansible的部署进行配置、此外、通常还建议使用其他几种设置来优化性能/行为、其中包括：

- 将全局缓存块大小调整为32 KiB、并将基于需求的缓存刷新调整为80%。
- 禁用自动加载平衡(确保控制器卷分配保持预期状态)。
- 启用读取缓存和禁用预读缓存。
- 通过镜像启用写入缓存并要求使用电池备份、以便在块节点控制器发生故障时保持缓存不变。
- 指定驱动器分配给卷组的顺序、在可用驱动器通道之间平衡I/O。

大容量组件

标准BeeGFS解决方案 设计充分考虑了高性能工作负载。寻求大容量使用情形的客户应观察此处所述的设计和性能特征的差异。

硬件和软件配置

大容量组件的硬件和软件配置是标准配置、但EF600控制器应更换为EF300控制器、并可选择在1到7个IOM扩展托盘之间连接、每个存储阵列使用60个驱动器。每个组件总共有2到14个扩展托盘。

部署高容量构建块设计的客户可能仅使用基础构建块模式配置、该配置包含每个节点的BeeGFS管理、元数据和存储服务。为了节省成本、高容量存储节点应在EF300控制器机箱中的NVMe驱动器上配置元数据卷、并为扩展托盘中的NL-SAS驱动器配置存储卷。

[]

规模估算准则

这些规模估算准则假定高容量组件在基础EF300机箱中配置了一个2+2 NVMe SSD卷组用于元数据、并在每个IOM扩展托盘中配置了6个8+2 NL-SAS卷组用于存储。

驱动器大小(容量 HD D)	每个 BB 的容量(1个托盘)	每个 BB 的容量(2个托盘)	每个 BB 的容量(3个托盘)	每个 BB 的容量(4个托盘)
4 TB	439 TB	878 TB	1317 TB	1756 TB
8 TB	878 TB	1756 TB	2634 TB	3512 TB
10 TB	1097 TB	2195 TB	3292 TB	4390 TB
12 TB	1317 TB	2634 TB	3951 TB	5268 TB
16 TB	1756 TB	3512 TB	5268 TB	7024 TB
18 TB	1975-TB	3951 TB	5927 TB	7902 TB

部署解决方案

部署概述

NetApp上的BeeGFS可以通过将Ansient与NetApp的BeeGFS组件设计结合使用来部署到经过验证的文件和块节点。

Ansible集合和角色

基于NetApp的BeeGFS解决方案使用Andsient进行部署、该引擎是一个可自动执行应用程序部署的常用IT自动化引擎。Ansient使用一系列文件(统称为清单)、这些文件用于为您要部署的BeeGFS文件系统建模。

利用NetApp等公司的Ansilt，可以使用在Ansilt Galaxy上提供的集合来扩展内置功能(请参见 ["NetApp E系列BeeGFS集合"](#))。集合包括执行特定功能或任务(例如创建E系列卷)的模块以及可以调用多个模块和其他角色的角色。这种自动化方法可减少部署BeeGFS文件系统和底层HA集群所需的时间。此外、它还简化了集群和BeeGFS文件系统的维护和扩展。

有关其他详细信息，请参见 ["了解Ansible清单"](#)。



由于在NetApp解决方案 上部署BeeGFS涉及许多步骤、因此NetApp不支持手动部署解决方案。

BeeGFS组件的配置配置文件

部署过程包括以下配置文件：

- 一个基础组件、其中包括管理、元数据和存储服务。

- 包含元数据和存储服务的第二个组件。
- 仅包含存储服务的第三个组件。

这些配置文件展示了NetApp BeeGFS组件的完整建议配置文件。对于每个部署、元数据和存储组件或仅存储服务组件的数量可能会因容量和性能要求而异。

部署步骤概述

部署涉及以下高级任务：

硬件部署

1. 以物理方式组装每个组件。
2. 机架和缆线硬件。有关详细过程、请参见 ["部署硬件"](#)。

软件部署

1. ["设置文件和块节点"](#)。
 - 在文件节点上配置BMC IP
 - 安装受支持的操作系统并在文件节点上配置管理网络
 - 在块节点上配置管理IP
2. ["设置Ansible控制节点"](#)。
3. ["调整系统性能设置"](#)。
4. ["创建Ansible清单"](#)。
5. ["为BeeGFS组件定义Ansible清单"](#)。
6. ["使用Ansible部署BeeGFS"](#)。
7. ["配置BeeGFS客户端"](#)。

部署过程包括几个示例、其中需要将文本复制到文件。密切关注任何以“#”或“/”字符表示的内联注释，其中的内容应该或可以针对特定部署进行修改。例如：



```
`beegfs_ha_ntp_server_pools: # THIS IS AN EXAMPLE OF A COMMENT!
  - "pool 0.pool.ntp.org iburst maxsources 3"
  - "pool 1.pool.ntp.org iburst maxsources 3"``
```

部署建议各不相同的衍生架构：

- ["高容量组件"](#)

了解Ansible清单

在开始部署之前、请熟悉如何配置和使用Ansient来部署基于NetApp的BeeGFS解决方案。

此目录结构是一个目录结构、其中列出了要在其中部署BeeGFS文件系统的文件和块节点。它包括用于描述所需BeeGFS文件系统的主机、组和变量。需要将该清单存储在可访问用于运行可编阅指南的文件和块节点的任何

计算机上的可编阅控制节点上。可从下载样本清单 ["NetApp E系列BeeGFS GitHub"](#)。

Ansible模块和角色

要应用在"Ands还是"清单中描述的配置，请使用NetApp E系列的"Ands还是"集合中提供的各种"Ands还是"模块和角色(可从获取 ["NetApp E系列BeeGFS GitHub"](#))来部署端到端解决方案。

NetApp E系列Ansible资料集中的每个角色都是在NetApp解决方案 上完整地端到端部署BeeGFS。这些角色使用NetApp E系列SANtricity、主机和BeeGFS集合、通过这些集合、您可以使用HA (高可用性)配置BeeGFS文件系统。然后、您可以配置和映射存储、并确保集群存储已准备就绪、可以使用。

虽然角色附带了深入的文档、但部署过程介绍了如何使用角色使用第二代BeeGFS组件设计部署经验证的NetApp架构。



尽管部署步骤会尝试提供足够详细的信息、以便事先使用Ansible并不是前提条件、但您应该对Ansible及相关术语有所熟悉。

BeeGFS HA集群的清单布局

使用Ansient清单结构定义BeeGFS HA集群。

任何具有以往的Andsware经验的人员都应了解BeeGFS HA角色可通过一种自定义方法来发现适用于每个主机的变量(或事实)。这种设计简化了构建可在多台服务器上运行的资源的Ans维修 清单。

一个"Ands还是"And"清单通常由和 `group_vars`` 中的文件以及一个 ``inventory.yml`` 将主机分配给特定组 (可能还会将这些组分配给其他组) 的文件组成 ``host_vars``。



请勿使用本小节中的内容创建任何文件、本小节仅用作示例。

尽管此配置是根据配置配置文件预先确定的、但您应大致了解如何将所有内容作为Ansible清单进行布局、如下所示：

```

# BeeGFS HA (High Availability) cluster inventory.
all:
  children:
    # Ansible group representing all block nodes:
    eseries_storage_systems:
      hosts:
        netapp01:
        netapp02:
    # Ansible group representing all file nodes:
    ha_cluster:
      children:
        meta_01: # Group representing a metadata service with ID 01.
          hosts:
            beegfs_01: # This service is preferred on the first file
node.
                        beegfs_02: # And can failover to the second file node.
        meta_02: # Group representing a metadata service with ID 02.
          hosts:
            beegfs_02: # This service is preferred on the second file
node.
                        beegfs_01: # And can failover to the first file node.

```

对于每个服务、将在`group\_vars`下创建一个附加文件、用于描述其配置:

```
# meta_01 - BeeGFS HA Metadata Resource Group
beegfs_ha_beegfs_meta_conf_resource_group_options:
  connMetaPortTCP: 8015
  connMetaPortUDP: 8015
  tuneBindToNumaZone: 0
floating_ips:
  - i1b: <IP>/<SUBNET_MASK>
  - i2b: <IP>/<SUBNET_MASK>
# Type of BeeGFS service the HA resource group will manage.
beegfs_service: metadata # Choices: management, metadata, storage.
# What block node should be used to create a volume for this service:
beegfs_targets:
  netapp01:
    eseries_storage_pool_configuration:
      - name: beegfs_m1_m2_m5_m6
        raid_level: raid1
        criteria_drive_count: 4
        common_volume_configuration:
          segment_size_kb: 128
        volumes:
          - size: 21.25
            owning_controller: A
```

通过此布局、可以在一个位置为每个资源定义BeeGFS服务、网络和存储配置。在后台、BeeGFS角色会根据此清单结构为每个文件和块节点聚合所需的配置。



系统会根据组名称自动配置每个服务的BeeGFS数字和字符串节点ID。因此、除了组名称唯一的一般可逆要求之外、表示BeeGFS服务的组必须以该组所代表的BeeGFS服务类型唯一的数字结尾。例如、允许使用meta_01和stor_01、但不允许使用metadata_01和metada_01。

查看最佳实践

在NetApp解决方案 上部署BeeGFS时、请遵循最佳实践准则。

标准约定

在物理组装和创建Ansible清单文件时、请遵循以下标准约定(有关详细信息、请参见 ["创建Ansible清单"](#))。

- 文件节点主机名按顺序编号(h01-HN)、机架顶部的数字较低、底部的数字较高。

例如、命名约定 [location][row][rack]hN 如下所示： beegfs_01。

- 每个块节点由两个存储控制器组成、每个控制器都有自己的主机名。

存储阵列名称用于在Ansible清单中引用整个块存储系统。存储阵列名称应按顺序编号(A01 - A)、各个控制器的主机名均根据该命名约定派生。

例如，名为的块节点 `ictad22a01` 通常可以为每个控制器配置主机名，如 `ictad22a01-a` 和 `ictad22a01-b`，但在Ans得 清单中称为 `netapp_01`。

- 同一构建块中的文件和块节点共享相同的编号方案、并在机架中彼此相邻、两个文件节点位于顶部、两个块节点位于其正下方。

例如、在第一个构建块中、文件节点`h01`和`h02`都直接连接到块节点`A01`和`A02`。主机名依次为`h01`、`h02`、`A01`和`A02`。

- 组件会根据主机名按顺序进行安装、因此、编号较低的主机名位于机架顶部、编号较高的主机名位于底部。

其目的是最大限度地缩短连接到机架交换机顶部的缆线长度、并定义标准部署实践以简化故障排除。对于因机架稳定性问题而不允许这样做的数据中心、当然也允许使用相反方法、即从下往上填充机架。

InfiniBand存储网络配置

每个文件节点上一半的InfiniBand端口用于直接连接到块节点。另一半连接到InfiniBand交换机、并用于BeeGFS客户端-服务器连接。在确定用于BeeGFS客户端和服务器的IPoIB子网的大小时、您必须考虑计算/GPU集群和BeeGFS文件系统的预期增长。如果您必须偏离建议的IP范围、请注意、一个构建块中的每个直接连接都有一个唯一的子网、并且与用于客户端-服务器连接的子网没有重叠。

直接连接

每个构建块中的文件和块节点始终使用下表中的IP进行直接连接。



此寻址方案遵循以下规则：第三个八位字节始终为奇数或偶数、具体取决于文件节点是奇数还是偶数。

文件节点	IB端口	IP 地址	块节点	IB端口	物理IP	虚拟IP
奇数(H1)	i1a	192.168.1.10	奇数(C1)	2a.	192.168.1.100	192.168.1.101
奇数(H1)	i2a.	192.168.3.10	奇数(C1)	2a.	192.168.3.100	192.168.3.101
奇数(H1)	i3a	192.168.5.10	偶数(C2)	2a.	192.168.5.100	192.168.5.101
奇数(H1)	i4a	192.168.7.10	偶数(C2)	2a.	192.168.7.100	192.168.7.101
偶数(H2)	i1a	192.168.2.10	奇数(C1)	2亿	192.168.2.100	192.168.2.101
偶数(H2)	i2a.	192.168.4.10	奇数(C1)	2亿	192.168.4.100	192.168.4.101
偶数(H2)	i3a	192.168.6.10	偶数(C2)	2亿	192.168.6.100	192.168.6.101
偶数(H2)	i4a	192.168.8.10	偶数(C2)	2亿	192.168.8.100	192.168.8.101

BeeGFS客户端-服务器IPoIB寻址方案

每个文件节点都运行多个BeeGFS服务器服务(管理、元数据或存储)。为了使每个服务能够独立故障转移到另一个文件节点、每个服务都配置了唯一的IP地址、这些地址可以在两个节点之间浮动(有时称为逻辑接口或LIF)。

此部署虽然不是必需的、但假定这些连接正在使用以下IPoIB子网范围、并定义了一个标准寻址方案、该方案将应用以下规则：

- 根据文件节点InfiniBand端口是奇数还是奇数、第二个八位字节始终是奇数甚至偶数。

- BeeGFS集群IP始终为`xxx.127.100.yyy`或`xxx.128.100.yyy`。



除了用于带内操作系统管理的接口之外、Corosync还可以使用其他接口进行集群检测和同步。这样可以确保丢失一个接口不会导致整个集群关闭。

- BeeGFS管理服务始终为`xxx.yyy.101.0`或`xxx.yyy.102.0`。
- BeeGFS元数据服务始终位于`xxx.yyy.101.zzz`或`xxx.yyy.102.zzz`。
- BeeGFS存储服务始终位于`xxx.yyy.103.zzz`或`xxx.yyy.104.zzz`。
- 范围`100.xxx.1.1`到`100.xxx.99.255`的地址是为客户端预留的。

IPoIB单子网寻址方案

鉴于中列出的优势，本部署指南将使用单个子网模式 "软件架构"。

子网: **100.127.0.0/16**

下表提供了单个子网的范围: 100.127.0.0/16。

目的	InfiniBand端口	IP地址或范围
BeeGFS集群IP	i1b或i4b	100.127.100.1 - 100.127.100.255
BeeGFS管理	i1b	100.127.101.0
	i2b.	100.127.102.0
BeeGFS元数据	i1b或i3b	100.127.101.1 - 100.127.101.255
	i2b或i4b	100.127.102.1 - 100.127.102.255
BeeGFS存储	i1b或i3b	100.127.103.1 - 100.127.103.255
	i2b或i4b	100.127.104.1 - 100.127.104.255
BeeGFS客户端	(因客户端而异)	100.127.1.1 - 100.127.99.255

IPoIB双子网寻址方案

不再建议使用双子网编址方案、但仍可实施。有关建议的双子网方案、请参见下表。

子网A: **100.127.0.0/16**

下表提供了子网A的范围: 100.127.0.0/16。

目的	InfiniBand端口	IP地址或范围
BeeGFS集群IP	i1b	100.127.100.1 - 100.127.100.255
BeeGFS管理	i1b	100.127.101.0
BeeGFS元数据	i1b或i3b	100.127.101.1 - 100.127.101.255
BeeGFS存储	i1b或i3b	100.127.103.1 - 100.127.103.255
BeeGFS客户端	(因客户端而异)	100.127.1.1 - 100.127.99.255

子网B: **100.128.0.0/16**

下表提供了子网B的范围：100.128.0.0/16。

目的	InfiniBand端口	IP地址或范围
BeeGFS集群IP	i4b.	100.128.100.1 - 100.128.100.255
BeeGFS管理	i2b.	100.128.102.0
BeeGFS元数据	i2b或i4b	100.128.102.1 - 100.128.102.255
BeeGFS存储	i2b或i4b	100.128.104.1 - 100.128.104.255
BeeGFS客户端	(因客户端而异)	100.128.1.1 - 100.128.99.255



并非上述范围内的所有IP都用于此经过NetApp验证的架构。它们展示了如何预先分配IP地址、以便使用一致的IP寻址方案轻松扩展文件系统。在此方案中、BeeGFS文件节点和服务ID与已知IP范围中的第四个八位字节相对应。如果需要、文件系统可以扩展到255个节点或服务以上。

部署硬件

每个组件都包含两个经过验证的x86文件节点、这些文件节点使用HDR (200 GB) InfiniBand缆线直接连接到两个块节点。



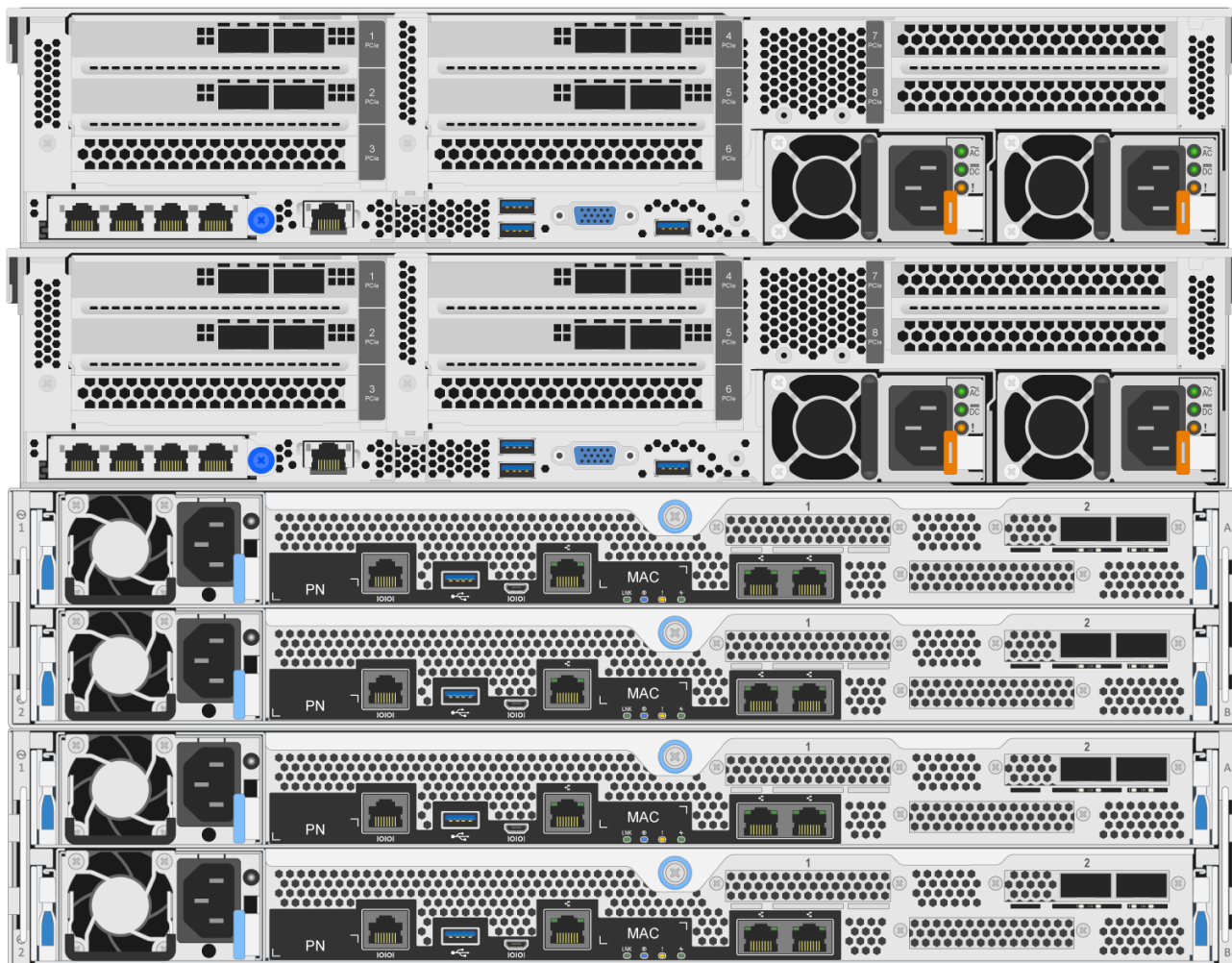
要在故障转移集群中建立仲裁、至少需要两个组件。双节点集群存在一些限制、可能会阻止成功进行故障转移。您可以通过将第三个设备整合为Tiebreaker 机来配置双节点集群；但是、本文档不会介绍这种设计。

除非另有说明、否则以下步骤对于集群中的每个组件都是相同的、无论该组件是同时用于运行BeeGFS元数据和存储服务、还是仅用于运行存储服务。

步骤

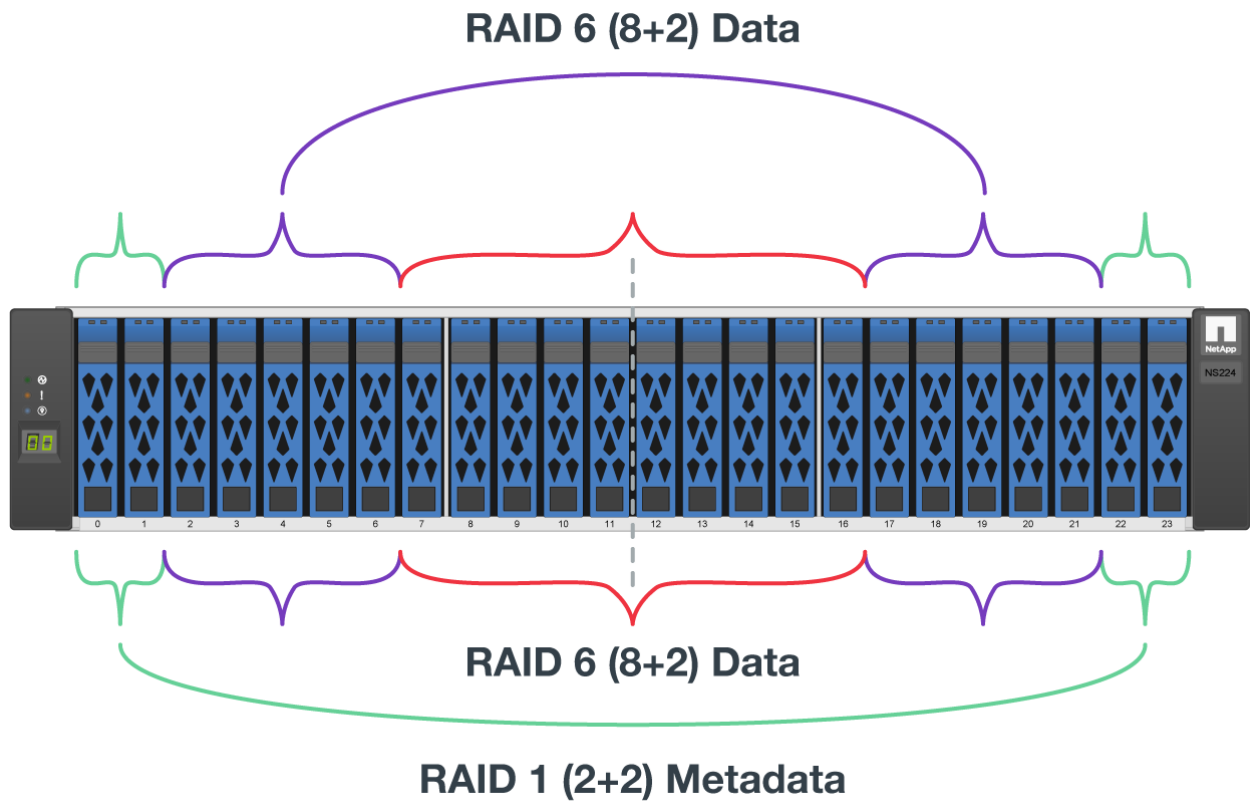
1. 使用中指定的型号为每个BeeGFS文件节点设置四个主机通道适配器(HCAS) "[技术要求](#)"。根据以下规格将HCA插入文件节点的PCIe插槽：
 - *联想凌科系统SR6SR V3服务器：*使用PCIe插槽1、2、4和5。
 - *Lenovo ThinkSystem SR6SR服务器：*使用PCIe插槽2、3、5和6。
2. 为每个BeeGFS块节点配置一个双端口200 GB主机接口卡(HIC)、并在其两个存储控制器中的每个控制器中安装HIC。

将组件装入机架、使两个BeeGFS文件节点位于BeeGFS块节点上方。下图显示了使用Lenovo ThinkSystem SR点 来作为文件节点的BeeGFS组件的正确硬件配置(后视图)。

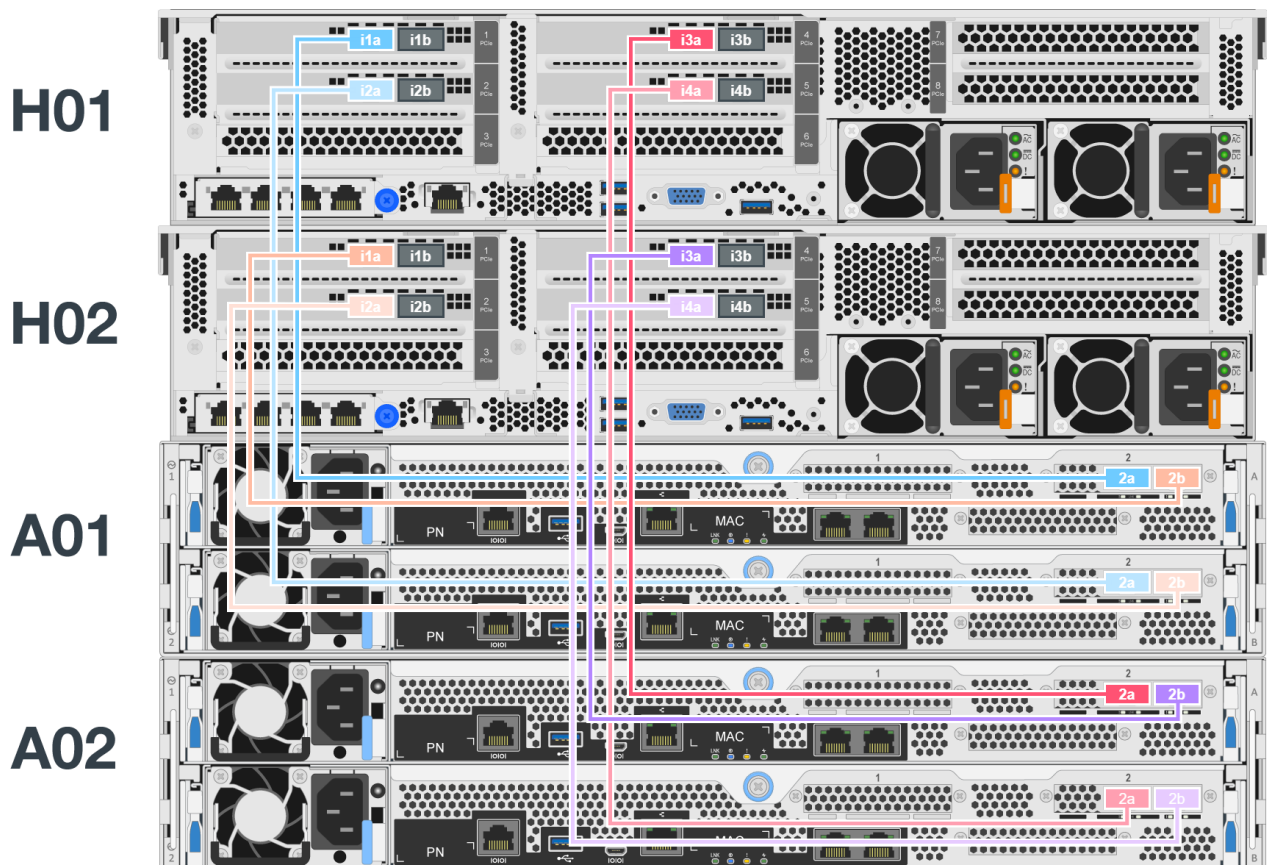


用于生产用例的电源配置通常应使用冗余PSU。

3. 如果需要、请在每个BeeGFS块节点中安装驱动器。
 - a. 如果要使用此组件运行BeeGFS元数据和存储服务、而将较小的驱动器用于元数据卷、请验证这些驱动器是否已填充到最外的驱动器插槽中、如下图所示。
 - b. 对于所有组件配置、如果驱动器机箱未完全填充、请确保在插槽0–11和12–23中填充相同数量的驱动器、以获得最佳性能。



4. 使用连接块节点和文件节点 "1米InfiniBand HDR 200 GB直连铜缆", 使它们与下图所示的拓扑相匹配。



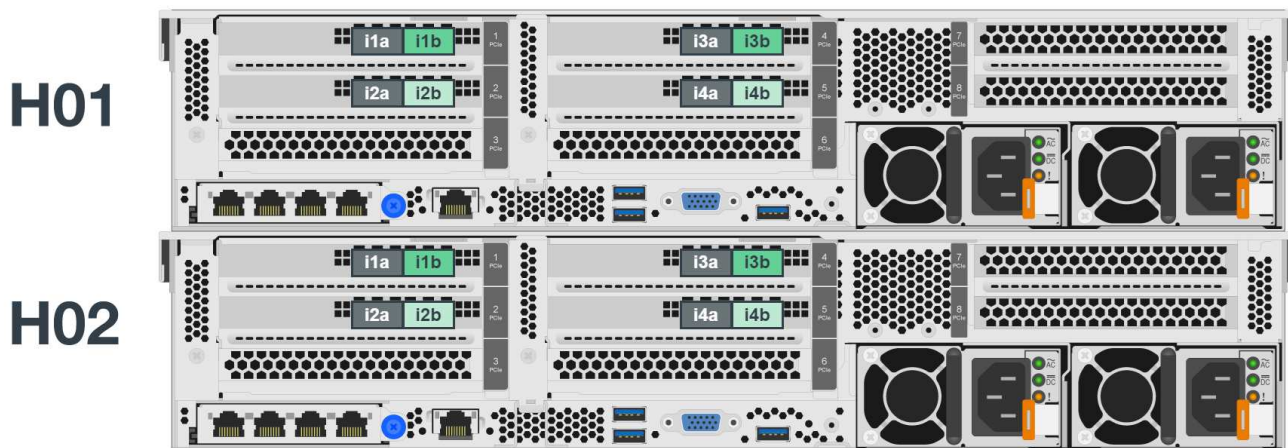


多个组件中的节点不会直接连接。每个组件都应视为一个独立单元、并且各个组件之间的所有通信都通过网络交换机进行。

5. 使用特定于InfiniBand存储交换机的将文件节点上的其余InfiniBand端口连接到存储网络的InfiniBand交换机"2米InfiniBand缆线"。

在使用分路器缆线将存储交换机连接到文件节点时、应从交换机分支出一根缆线、并连接到浅绿色所示的端口。另一根分离器电缆应从交换机分支出来、并连接到深绿色所示的端口。

此外、对于具有冗余交换机的存储网络、浅绿色端口应连接到一个交换机、而深绿色端口应连接到另一个交换机。



6. 根据需要、按照相同的布线准则组装其他组件。



可部署在一个机架中的组件总数取决于每个站点的可用电源和散热。

部署软件

设置文件节点和块节点

虽然大多数软件配置任务都是使用NetApp提供的Ansible集合自动执行的、但您必须在每个服务器的基板管理控制器(Baseboard Management Controller、BMC)上配置网络、并在每个控制器上配置管理端口。

设置文件节点

1. 在每个服务器的基板管理控制器(BMC)上配置网络连接。

要了解如何为经验证的联想SR点 对V3文件节点配置网络连接, 请参阅 ["联想ThinkSystem文档"](#)。



基板管理控制器(Baseboard Management Controller、BMC)有时也称为服务处理器、它是内置于各种服务器平台中的带外管理功能的通用名称、即使操作系统未安装或无法访问、也可以提供远程访问。供应商通常会使用自己的品牌推广此功能。例如、在联想SR665上、BMC称为_Lenovo Xicity1控制器(XCC)_。

2. 配置系统设置以实现最高性能。

您可以使用UEFI设置(以前称为BIOS)或许多BMC提供的Redfish API配置系统设置。系统设置因用作文件节点的服务器型号而异。

要了解如何配置已验证的 Lenovo SR665 V3 文件节点的系统设置，请参阅["调整系统性能设置"](#)。

3. 安装 Red Hat Enterprise Linux (RHEL) 9.4 并配置用于管理操作系统的主机名和网络端口，包括来自 Ansible 控制节点的 SSH 连接。

此时、请勿在任何InfiniBand端口上配置IP。



尽管严格要求不高、但后续章节假定主机名按顺序编号(例如H1-HN)、并提及应在奇数主机上完成的任务、而不是偶数主机上完成的任务。

4. 使用 Red Hat Subscription Manager 注册并订阅系统，以允许从官方 Red Hat 存储库安装所需的软件包，并将更新限制在受支持的 Red Hat 版本上： `subscription-manager release --set=9.4`。有关说明，请参阅["如何注册和订阅RHEL系统"](#)和["如何限制更新"](#)。
5. 启用包含高可用性所需软件包的Red Hat存储库。

```
subscription-manager repo-override --repo=rhel-9-for-x86_64
-highavailability-rpms --add=enabled:1
```

6. 使用["更新文件节点适配器固件"](#)指南将所有HCA固件更新到中建议的版本["技术要求"](#)。

设置块节点

通过在每个控制器上配置管理端口来设置EF600块节点。

1. 在每个EF600控制器上配置管理端口。

有关配置端口的说明，请转到["E系列文档中心"](#)。

2. (可选)设置每个系统的存储阵列名称。

通过设置名称、可以更轻松地在后续章节中引用每个系统。有关设置阵列名称的说明，请转到["E系列文档中心"](#)。



尽管严格要求不严格、但后续主题假定存储阵列名称按顺序编号(例如C1 - CN)、并提及在奇数系统甚至编号系统上应完成的步骤。

调整文件节点系统设置以提高性能

为了最大程度地提高性能、我们建议在用作文件节点的服务器型号上配置系统设置。

系统设置因用作文件节点的服务器型号而异。本主题介绍如何为经过验证的联想ThinkSystem SR665服务器文件节点配置系统设置。

使用UEFI界面调整系统设置

Lenovo SR665 V3 服务器的系统固件包含许多可通过 UEFI 界面设置的调整参数。这些调整参数可能会影响服务器的运行方式以及服务器的性能。

在* UEFI设置>系统设置*下、调整以下系统设置：

操作模式菜单

系统设置	更改为
操作模式	自定义
cTDP	手动
cTDP手册	350
软件包电源限制	手动
效率模式	禁用
全局Cstate-Control	禁用
SOC P状态	P0
DF C状态	禁用
P状态	禁用
启用内存关闭	禁用
每个插槽的NUMA节点数	NPS1

设备和I/O端口菜单

系统设置	更改为
IOMMU	禁用

电源菜单

系统设置	更改为
PCIe电源制动器	禁用

处理器菜单

系统设置	更改为
全局C状态控制	禁用
DF C状态	禁用
SMT模式	禁用
CPPC	禁用

使用Redfish API调整系统设置

除了使用UEFI设置之外、您还可以使用Redfish API更改系统设置。

```
curl --request PATCH \
  --url https://<BMC_IP_ADDRESS>/redfish/v1/Systems/1/Bios/Pending \
  --user <BMC_USER>:<BMC- PASSWORD> \
  --header 'Content-Type: application/json' \
  --data '{
"Attributes": {
"OperatingModes_ChoseOperatingMode": "CustomMode",
"Processors_cTDP": "Manual",
"Processors_PackagePowerLimit": "Manual",
"Power_EfficiencyMode": "Disable",
"Processors_GlobalC_stateControl": "Disable",
"Processors_SOCP_states": "P0",
"Processors_DFC_States": "Disable",
"Processors_P_State": "Disable",
"Memory_MemoryPowerDownEnable": "Disable",
"DevicesandIOPorts_IOMMU": "Disable",
"Power_PCIePowerBrake": "Disable",
"Processors_GlobalC_stateControl": "Disable",
"Processors_DFC_States": "Disable",
"Processors_SMTMode": "Disable",
"Processors_CPPC": "Disable",
"Memory_NUMANodesperSocket": "NPS1"
}
}
'
```

有关Redfish模式的详细信息、请参见 ["DMTF网站"](#)。

设置Ansible控制节点

要设置可扩展控制节点、您必须指定一台虚拟机或物理机、以便能够通过网络访问为基于NetApp的BeeGFS解决方案部署的所有文件和块节点。

查看、["技术要求"](#)了解建议的软件包版本列表。以下步骤已在Ubuntu 22.04上进行了测试。有关首选Linux发行版的特定步骤，请参见 ["Ansible文档"](#)。

1. 从您的Ansible控制节点、安装以下Python和Python虚拟环境软件包。

```
sudo apt-get install python3 python3-pip python3-setuptools python3.10-venv
```

2. 创建 Python 虚拟环境。

```
python3 -m venv ~/pyenv
```

3. 激活虚拟环境。

```
source ~/pyenv/bin/activate
```

4. 在激活的虚拟环境中安装所需的Python软件包。

```
pip install ansible netaddr cryptography passlib
```

5. 使用Ansible Galaxy安装BeeGFS集合。

```
ansible-galaxy collection install netapp_eseries.beegfs
```

6. 验证已安装的Ansible、Python版本和BeeGFS集合是否与匹配["技术要求"](#)。

```
ansible --version
ansible-galaxy collection list netapp_eseries.beegfs
```

7. 设置无密码SSH、以允许Ansible从Ansible控制节点访问远程BeeGFS文件节点。

- a. 如果需要、在Ansible控制节点上生成一对公共密钥。

```
ssh-keygen
```

- b. 为每个文件节点设置无密码SSH。

```
ssh-copy-id <ip_or_hostname>
```



Do * not* set up passwordless SSH to the block nodes. 这既不受支持、也不是必需的。

创建Ansible清单

要定义文件和块节点的配置、请创建一个Ansible清单、用于表示要部署的BeeGFS文件系统。清单包括描述所需BeeGFS文件系统的主机、组和变量。

第1步：定义所有组件的配置

定义适用场景 所有组件的配置、无论您可以单独应用于这些组件的配置文件如何。

开始之前

- 为您的部署选择子网编址方案。由于中列出的优势 ["软件架构"](#)，建议使用单个子网编址方案。

步骤

1. 在Ansible控制节点上、确定要用于存储Ansible清单和攻略手册文件的目录。

除非另有说明、否则在此步骤中创建的所有文件和目录以及后续步骤均会相对于此目录创建。

2. 创建以下子目录：

host_vars

group_vars

软件包

3. 为集群密码创建子目录，并通过使用Ans可 存储对文件进行加密来保护文件(请参见 ["使用Ansible Vault加密内容"](#))：
 - a. 创建子目录 group_vars/all。
 - b. 在目录中 group_vars/all，创建一个标记为的密码文件 passwords.yml。
 - c. 使用以下内容填充 passwords.yml file、根据您的配置替换所有用户名和密码参数：

```
# Credentials for storage system's admin password
eseries_password: <PASSWORD>

# Credentials for BeeGFS file nodes
ssh_ha_user: <USERNAME>
ssh_ha_become_pass: <PASSWORD>

# Credentials for HA cluster
ha_cluster_username: <USERNAME>
ha_cluster_password: <PASSWORD>
ha_cluster_password_sha512_salt: randomSalt

# Credentials for fencing agents
# OPTION 1: If using APC Power Distribution Units (PDUs) for fencing:
# Credentials for APC PDUs.
apc_username: <USERNAME>
apc_password: <PASSWORD>

# OPTION 2: If using the Redfish APIs provided by the Lenovo XCC (and
other BMCs) for fencing:
# Credentials for XCC/BMC of BeeGFS file nodes
bmc_username: <USERNAME>
bmc_password: <PASSWORD>
```

d. 根据提示运行 `ansible-vault encrypt passwords.yml` 并设置存储密码。

第2步：为单个文件和块节点定义配置

定义适用场景 单个文件节点和单个组件节点的配置。

1. 在 `host\_vars/` 下、为名为 `<HOSTNAME>.yml` 的每个 BeeGFS 文件节点创建一个文件、其中包含以下内容、请特别注意有关 BeeGFS 集群 IP 和以奇数与偶数结尾的主机名要填充的内容的注释。

最初、文件节点接口名称与此处列出的名称匹配(例如 ib0 或 ibs1f0)。这些自定义名称在中进行配置 [\[第4步：定义应用于所有文件节点的配置\]](#)。

```

ansible_host: "<MANAGEMENT_IP>"
eseries_ipoib_interfaces: # Used to configure BeeGFS cluster IP
addresses.
  - name: ilb
    address: 100.127.100. <NUMBER_FROM_HOSTNAME>/16
  - name: i4b
    address: 100.127.100. <NUMBER_FROM_HOSTNAME>/16
beegfs_ha_cluster_node_ips:
  - <MANAGEMENT_IP>
  - <i1b_BEEGFS_CLUSTER_IP>
  - <i4b_BEEGFS_CLUSTER_IP>
# NVMe over InfiniBand storage communication protocol information
# For odd numbered file nodes (i.e., h01, h03, ..):
eseries_nvme_ib_interfaces:
  - name: i1a
    address: 192.168.1.10/24
    configure: true
  - name: i2a
    address: 192.168.3.10/24
    configure: true
  - name: i3a
    address: 192.168.5.10/24
    configure: true
  - name: i4a
    address: 192.168.7.10/24
    configure: true
# For even numbered file nodes (i.e., h02, h04, ..):
# NVMe over InfiniBand storage communication protocol information
eseries_nvme_ib_interfaces:
  - name: i1a
    address: 192.168.2.10/24
    configure: true
  - name: i2a
    address: 192.168.4.10/24
    configure: true
  - name: i3a
    address: 192.168.6.10/24
    configure: true
  - name: i4a
    address: 192.168.8.10/24
    configure: true

```



如果您已部署BeeGFS集群、则必须先停止此集群、然后才能添加或更改静态配置的IP地址、包括用于NVMe/IB的集群IP和IP。这是必需的、以便这些更改能够正确生效、并且不会中断集群操作。

2. 在`host\_vars/`下、为名为`<HOSTNAME>.yml`的每个BeeGFS块节点创建一个文件、并使用以下内容填充该文件。

请特别注意有关以奇数与偶数结尾的存储阵列名称要填充的内容的注释。

对于每个块节点、创建一个文件并为两个控制器(通常为A)之一指定`<management\_IP>`。

```
eseries_system_name: <STORAGE_ARRAY_NAME>
eseries_system_api_url: https://<MANAGEMENT_IP>:8443/devmgr/v2/
eseries_initiator_protocol: nvme_ib
# For odd numbered block nodes (i.e., a01, a03, ..):
eseries_controller_nvme_ib_port:
  controller_a:
    - 192.168.1.101
    - 192.168.2.101
    - 192.168.1.100
    - 192.168.2.100
  controller_b:
    - 192.168.3.101
    - 192.168.4.101
    - 192.168.3.100
    - 192.168.4.100
# For even numbered block nodes (i.e., a02, a04, ..):
eseries_controller_nvme_ib_port:
  controller_a:
    - 192.168.5.101
    - 192.168.6.101
    - 192.168.5.100
    - 192.168.6.100
  controller_b:
    - 192.168.7.101
    - 192.168.8.101
    - 192.168.7.100
    - 192.168.8.100
```

第3步：定义应用于所有文件和块节点的配置

您可以使用与组对应的文件名在`group\_vars`下定义一组主机的通用配置。这样可以防止在多个位置重复使用共享配置。

关于此任务

主机可以位于多个组中、运行时、Ansible会根据特定主机的可变优先级规则选择哪些变量适用于该主机。(有关这些规则的详细信息、请参见Ansible文档["使用变量"](#))

主机到组分配在实际Ansible清单文件中进行定义、该文件在该操作步骤 接近末尾时创建。

步骤

在Ansible中、可以在名为`All`的组中定义要应用于所有主机的任何配置。使用以下内容创建文件`group\_vars/all.yml`：

```
ansible_python_interpreter: /usr/bin/python3
beegfs_ha_ntp_server_pools: # Modify the NTP server addresses if
desired.
- "pool 0.pool.ntp.org iburst maxsources 3"
- "pool 1.pool.ntp.org iburst maxsources 3"
```

第4步：定义应用于所有文件节点的配置

文件节点的共享配置在名为`ha\_cluster`的组中定义。本节中的步骤将构建应包含在`group\_vars/ha\_cluster.yml`文件中的配置。

步骤

1. 在文件顶部、定义默认值、包括用作文件节点上的`sUdo`用户的密码。

```
### ha_cluster Ansible group inventory file.
# Place all default/common variables for BeeGFS HA cluster resources
below.
### Cluster node defaults
ansible_ssh_user: {{ ssh_ha_user }}
ansible_become_password: {{ ssh_ha_become_pass }}
eseries_ipoib_default_hook_templates:
- 99-multihoming.j2 # This is required for single subnet
deployments, where static IPs containing multiple IB ports are in the
same IPoIB subnet. i.e: cluster IPs, multirail, single subnet, etc.
# If the following options are specified, then Ansible will
automatically reboot nodes when necessary for changes to take effect:
eseries_common_allow_host_reboot: true
eseries_common_reboot_test_command: "! systemctl status
eseries_nvme_ib.service || systemctl --state=exited | grep
eseries_nvme_ib.service"
eseries_ib_opensm_options:
  virt_enabled: "2"
  virt_max_ports_in_process: "0"
```



如果`ansible\_ssh\_user`已经是`root`，则可以选择省略，
`ansible\_become\_password`并在运行该播放手册时指定`--ask-become-pass`选项。

2. (可选)配置高可用性(HA)集群的名称、并为集群内通信指定用户。

如果要修改专用IP寻址方案、则还必须更新默认值`beegfs\_ha\_mgmt\_float\_IP`。这必须与您稍后为BeeGFS管理资源组配置的内容匹配。

使用`beegfs\_ha\_alert\_email\_list`指定一个或多个应接收集群事件警报的电子邮件。

```
### Cluster information
beegfs_ha_firewall_configure: True
eseries_beegfs_ha_disable_selinux: True
eseries_selinux_state: disabled
# The following variables should be adjusted depending on the desired
configuration:
beegfs_ha_cluster_name: hacluster                # BeeGFS HA cluster
name.
beegfs_ha_cluster_username: "{{ ha_cluster_username }}" # Parameter for
BeeGFS HA cluster username in the passwords file.
beegfs_ha_cluster_password: "{{ ha_cluster_password }}" # Parameter for
BeeGFS HA cluster username's password in the passwords file.
beegfs_ha_cluster_password_sha512_salt: "{{
ha_cluster_password_sha512_salt }}" # Parameter for BeeGFS HA cluster
username's password salt in the passwords file.
beegfs_ha_mgmtd_floating_ip: 100.127.101.0        # BeeGFS management
service IP address.
# Email Alerts Configuration
beegfs_ha_enable_alerts: True
beegfs_ha_alert_email_list: ["email@example.com"] # E-mail recipient
list for notifications when BeeGFS HA resources change or fail. Often a
distribution list for the team responsible for managing the cluster.
beegfs_ha_alert_conf_ha_group_options:
    mydomain: "example.com"
# The mydomain parameter specifies the local internet domain name. This
is optional when the cluster nodes have fully qualified hostnames (i.e.
host.example.com).
# Adjusting the following parameters is optional:
beegfs_ha_alert_timestamp_format: "%Y-%m-%d %H:%M:%S.%N" # %H:%M:%S.%N
beegfs_ha_alert_verbosity: 3
# 1) high-level node activity
# 3) high-level node activity + fencing action information + resources
(filter on X-monitor)
# 5) high-level node activity + fencing action information + resources
```



虽然`冗余`、但当您将BeeGFS文件系统扩展到单个HA集群之外时、`beegfs\_ha\_mgmtd\_floating\_ip`非常重要。部署后续HA集群时无需额外的BeeGFS管理服务、并指向第一个集群提供的管理服务。

3. 配置隔离代理。(有关详细信息, 请参见 ["在Red Hat High Availability集群中配置隔离"](#)。)以下输出显示了配置常见隔离代理的示例。选择以下选项之一。

在此步骤中、请注意:

- 默认情况下、隔离处于启用状态、但您需要配置隔离_agent_。
- 在`PCMK\_HOST\_MAP`或`PCMK\_HOST\_LIST`中指定的`<HOSTNAME>`必须与Ansible清单中的主机名相对应。
- 不支持在不使用隔离的情况下运行BeeGFS集群、尤其是在生产环境中。这在很大程度上是为了确保当BeeGFS服务(包括块设备等任何资源依赖关系)因问题描述 而发生故障转移时、不会存在多个节点并发访问导致文件系统损坏或其他不希望或意外行为的风险。如果必须禁用隔离、请参阅BeeGFS HA角色的入门指南中的一般说明、并在`ha\_cluster\_crm\_config\_options["stonith-enabled"]`中将`beegfs\_ha\_cluster.yml`设置为false。
- 可以使用多个节点级别的隔离设备、BeeGFS HA角色可以配置Red Hat HA软件包存储库中可用的任何隔离代理。如果可能、请使用通过不间断电源(UPS)或机架配电单元(rPDU)工作的隔离代理。因为在某些故障情形下、某些隔离代理(如基板管理控制器(BMC)或服务器中内置的其他无人值守设备)可能无法响应隔离请求。

```

### Fencing configuration:
# OPTION 1: To enable fencing using APC Power Distribution Units
(PDUs):
beegfs_ha_fencing_agents:
  fence_apc:
    - ipaddr: <PDU_IP_ADDRESS>
      login: "{{ apc_username }}" # Parameter for APC PDU username in
the passwords file.
      passwd: "{{ apc_password }}" # Parameter for APC PDU password in
the passwords file.
      pcmk_host_map:
"<HOSTNAME>:<PDU_PORT>,<PDU_PORT>;<HOSTNAME>:<PDU_PORT>,<PDU_PORT>"
# OPTION 2: To enable fencing using the Redfish APIs provided by the
Lenovo XCC (and other BMCs):
redfish: &redfish
  username: "{{ bmc_username }}" # Parameter for XCC/BMC username in
the passwords file.
  password: "{{ bmc_password }}" # Parameter for XCC/BMC password in
the passwords file.
  ssl_insecure: 1 # If a valid SSL certificate is not available
specify "1".
beegfs_ha_fencing_agents:
  fence_redfish:
    - pcmk_host_list: <HOSTNAME>
      ip: <BMC_IP>
      <<: *redfish
    - pcmk_host_list: <HOSTNAME>
      ip: <BMC_IP>
      <<: *redfish

# For details on configuring other fencing agents see
https://access.redhat.com/documentation/en-
us/red\_hat\_enterprise\_linux/9/html/configuring\_and\_managing\_high\_avai
lability\_clusters/assembly\_configuring-fencing-configuring-and-
managing-high-availability-clusters.

```

4. 在Linux操作系统中启用建议的性能调整。

虽然许多用户发现性能参数的默认设置通常运行良好、但您也可以选择更改特定工作负载的默认设置。因此、这些建议包含在BeeGFS角色中、但默认情况下不会启用、以确保用户了解应用于其文件系统的调整。

要启用性能调整、请指定：

```

### Performance Configuration:
beegfs_ha_enable_performance_tuning: True

```

5. (可选)您可以根据需要调整Linux操作系统中的性能调整参数。

有关可调整的可用调整参数的完整列表，请参见中BeeGFS HA角色的“性能调整默认值”部分 ["E系列BeeGFS GitHub站点"](#)。可以覆盖此文件或单个节点的文件中集群中所有节点的默认值 `host_vars`。

6. 要在块节点和文件节点之间实现完整的200GB/HDR连接、请使用NVIDIA开放式网络结构企业分发版(MLNR_OFED)中的开放式子网管理器(OpenSM)软件包。中列出的MLNR_OFED版本 ["文件节点要求"](#) 与建议的OpenSM软件包捆绑在一起。虽然支持使用Ansibly进行部署、但您必须先在所有文件节点上安装MLNR_OFED驱动程序。
- a. 在`group\_vars/ha\_cluster.yml`中填充以下参数(根据需要调整软件包):

```
### OpenSM package and configuration information
eseries_ib_opensm_options:
  virt_enabled: "2"
  virt_max_ports_in_process: "0"
```

7. 配置`udev`规则、以确保逻辑InfiniBand端口标识符与底层PCIe设备的映射一致。

`udev`规则对于用作BeeGFS文件节点的每个服务器平台的PCIe拓扑来说必须是唯一的。

对于已验证的文件节点、请使用以下值:

```
### Ensure Consistent Logical IB Port Numbering
# OPTION 1: Lenovo SR665 V3 PCIe address-to-logical IB port mapping:
eseries_ipoib_udev_rules:
  "0000:01:00.0": i1a
  "0000:01:00.1": i1b
  "0000:41:00.0": i2a
  "0000:41:00.1": i2b
  "0000:81:00.0": i3a
  "0000:81:00.1": i3b
  "0000:a1:00.0": i4a
  "0000:a1:00.1": i4b

# OPTION 2: Lenovo SR665 PCIe address-to-logical IB port mapping:
eseries_ipoib_udev_rules:
  "0000:41:00.0": i1a
  "0000:41:00.1": i1b
  "0000:01:00.0": i2a
  "0000:01:00.1": i2b
  "0000:a1:00.0": i3a
  "0000:a1:00.1": i3b
  "0000:81:00.0": i4a
  "0000:81:00.1": i4b
```

8. (可选)更新元数据目标选择算法。

```
beegfs_ha_beegfs_meta_conf_ha_group_options:
  tuneTargetChooser: randomrobin
```



在验证测试中、通常会使用`Randomrobin`来确保测试文件在性能基准测试期间均匀分布在所有BeeGFS存储目标上(有关基准测试的详细信息、请参见BeeGFS站点 "[对BeeGFS系统进行基准测试](#)")。在实际使用情况下、这可能会导致编号较低的目标达到发生原因、从而比编号较高的目标更快地达到填充速度。已显示省略`randomrobin`以及仅使用默认值`randomized`值可在仍利用所有可用目标的情况下提供良好的性能。

第5步：定义通用块节点的配置

块节点的共享配置在名为`Eseries\_storage\_systems`的组中定义。本节中的步骤构建了应包含在`group\_vars/eseries\_storage\_systems.yml`文件中的配置。

步骤

1. 将Ansible连接设置为local、提供系统密码、并指定是否应验证SSL证书。(通常、Ansible使用SSH连接到受管主机、但对于用作块节点的NetApp E系列存储系统、模块使用REST API进行通信。) 在文件顶部、添加以下内容：

```
### eseries_storage_systems Ansible group inventory file.
# Place all default/common variables for NetApp E-Series Storage Systems
here:
ansible_connection: local
eseries_system_password: {{ eseries_password }} # Parameter for E-Series
storage array password in the passwords file.
eseries_validate_certs: false
```

2. 为确保获得最佳性能、请在中安装为块节点列出的版本 "[技术要求](#)"。

从下载相应的文件 "[NetApp 支持站点](#)"。您可以手动升级它们、也可以将其包含在Ansible控制节点的`packages/`目录中、然后在`Esery\_storage\_systems.yml`中填充以下参数以使用Ansible进行升级：

```
# Firmware, NVSRAM, and Drive Firmware (modify the filenames as needed):
eseries_firmware_firmware: "packages/RCB_11.80GA_6000_64cc0ee3.dlp"
eseries_firmware_nvram: "packages/N6000-880834-D08.dlp"
```

3. 从下载并安装可用于块节点中安装的驱动器的最新驱动器固件 "[NetApp 支持站点](#)"。您可以手动升级它们、也可以将其包含在Ansul任`packages/`控制节点的目录中、然后在中填充以下参数
eseries_storage_systems.yml 以使用Ansul任 升级：

```
eseries_drive_firmware_firmware_list:
  - "packages/<FILENAME>.dlp"
eseries_drive_firmware_upgrade_drives_online: true
```



将`esery\_drive\_firmware\_upgrade\_drives\_online`设置为`false`可以加快升级速度、但在部署BeeGFS之前不应执行此操作。这是因为该设置要求在升级之前停止驱动器的所有I/O、以避免应用程序错误。尽管在配置卷之前执行联机驱动器固件升级仍很快、但我们建议您始终将此值设置为`true`以避免稍后出现问题。

4. 要优化性能、请对全局配置进行以下更改：

```
# Global Configuration Defaults
eseries_system_cache_block_size: 32768
eseries_system_cache_flush_threshold: 80
eseries_system_default_host_type: linux dm-mp
eseries_system_autoload_balance: disabled
eseries_system_host_connectivity_reporting: disabled
eseries_system_controller_shelf_id: 99 # Required.
```

5. 要确保最佳卷配置和行为、请指定以下参数：

```
# Storage Provisioning Defaults
eseries_volume_size_unit: pct
eseries_volume_read_cache_enable: true
eseries_volume_read_ahead_enable: false
eseries_volume_write_cache_enable: true
eseries_volume_write_cache_mirror_enable: true
eseries_volume_cache_without_batteries: false
eseries_storage_pool_usable_drives:
"99:0,99:23,99:1,99:22,99:2,99:21,99:3,99:20,99:4,99:19,99:5,99:18,99:6,
99:17,99:7,99:16,99:8,99:15,99:9,99:14,99:10,99:13,99:11,99:12"
```



为`E系列\_storage\_pool\_usable\_drives`指定的值特定于NetApp EF600块节点、并控制驱动器分配给新卷组的顺序。此顺序可确保每个组的I/O在后端驱动器通道之间均匀分布。

为BeeGFS组件定义Ansible清单

定义了常规Ansible清单结构后、为BeeGFS文件系统每个组件定义配置。

这些部署说明演示了如何部署文件系统、该文件系统由包括管理、元数据和存储服务在内的基础组件、包含元数据和存储服务的第二个组件以及仅存储的第三个组件组成。

这些步骤旨在显示一系列典型配置文件、您可以使用这些配置文件来配置NetApp BeeGFS组件、以满足整

个BeeGFS文件系统的要求。



在本节及后续章节中、根据需要进行调整、以构建表示要部署的BeeGFS文件系统的清单。特别是、请使用表示每个块或文件节点的Ansible主机名以及存储网络所需的IP寻址方案、以确保它可以扩展到BeeGFS文件节点和客户端的数量。

第1步：创建Ansible清单文件

步骤

1. 创建新的`inventory.yml`文件、然后插入以下参数、根据需要替换`Esery\_storage\_systems`下的主机、以表示部署中的块节点。这些名称应与`host\_vars/<filename>.yml`使用的名称相对应。

```
# BeeGFS HA (High Availability) cluster inventory.
all:
  children:
    # Ansible group representing all block nodes:
    eseries_storage_systems:
      hosts:
        netapp_01:
        netapp_02:
        netapp_03:
        netapp_04:
        netapp_05:
        netapp_06:
    # Ansible group representing all file nodes:
    ha_cluster:
      children:
```

在后续各节中、您将在`ha\_cluster`下创建其他Ansible组、这些组表示您要在集群中运行的BeeGFS服务。

第2步：为管理、元数据和存储组件配置清单

集群或基础构建块中的第一个组件必须包括BeeGFS管理服务以及元数据和存储服务：

步骤

1. 在`inventory.yml`中、在`ha\_cluster`下填充以下参数：子项`：

```
# beegfs_01/beegfs_02 HA Pair (mgmt/meta/storage building block):
mgmt:
  hosts:
    beegfs_01:
    beegfs_02:
meta_01:
  hosts:
    beegfs_01:
    beegfs_02:
```

```
stor_01:
  hosts:
    beegfs_01:
    beegfs_02:
meta_02:
  hosts:
    beegfs_01:
    beegfs_02:
stor_02:
  hosts:
    beegfs_01:
    beegfs_02:
meta_03:
  hosts:
    beegfs_01:
    beegfs_02:
stor_03:
  hosts:
    beegfs_01:
    beegfs_02:
meta_04:
  hosts:
    beegfs_01:
    beegfs_02:
stor_04:
  hosts:
    beegfs_01:
    beegfs_02:
meta_05:
  hosts:
    beegfs_02:
    beegfs_01:
stor_05:
  hosts:
    beegfs_02:
    beegfs_01:
meta_06:
  hosts:
    beegfs_02:
    beegfs_01:
stor_06:
  hosts:
    beegfs_02:
    beegfs_01:
meta_07:
  hosts:
```

```

        beegfs_02:
        beegfs_01:
stor_07:
  hosts:
    beegfs_02:
    beegfs_01:
meta_08:
  hosts:
    beegfs_02:
    beegfs_01:
stor_08:
  hosts:
    beegfs_02:
    beegfs_01:

```

2. 创建文件`group\_vars/mgmt.yml`并包含以下内容:

```

# mgmt - BeeGFS HA Management Resource Group
# OPTIONAL: Override default BeeGFS management configuration:
# beegfs_ha_beeGFS_mgmt_d_conf_resource_group_options:
# <beegfs-mgmt.conf:key>:<beegfs-mgmt.conf:value>
floating_ips:
  - i1b: 100.127.101.0/16
  - i2b: 100.127.102.0/16
beegfs_service: management
beegfs_targets:
  netapp_01:
    eseries_storage_pool_configuration:
      - name: beegfs_m1_m2_m5_m6
        raid_level: raid1
        criteria_drive_count: 4
        common_volume_configuration:
          segment_size_kb: 128
        volumes:
          - size: 1
            owning_controller: A

```

3. 在`group\_vars/`下、使用以下模板为资源组`mETA\_01`到`mETA\_08`创建文件、然后参考下表填写每个服务的占位值:

```
# meta_0X - BeeGFS HA Metadata Resource Group
beegfs_ha_beegfs_meta_conf_resource_group_options:
  connMetaPortTCP: <PORT>
  connMetaPortUDP: <PORT>
  tuneBindToNumaZone: <NUMA_ZONE>
floating_ips:
  - <PREFERRED PORT:IP/SUBNET> # Example: i1b:192.168.120.1/16
  - <SECONDARY PORT:IP/SUBNET>
beegfs_service: metadata
beegfs_targets:
  <BLOCK NODE>:
    eseries_storage_pool_configuration:
      - name: <STORAGE POOL>
        raid_level: raid1
        criteria_drive_count: 4
        common_volume_configuration:
          segment_size_kb: 128
        volumes:
          - size: 21.25 # SEE NOTE BELOW!
            owning_controller: <OWNING CONTROLLER>
```



卷大小以整个存储池(也称为卷组)的百分比形式指定。NetApp强烈建议您在每个池中保留一些可用容量、以便为SSD过度配置留出空间(有关详细信息、请参见 ["NetApp EF600阵列简介"](#))。存储池`beegfs\_m1\_m2\_m3\_m3\_m6`也会将池容量的1%分配给管理服务。因此、对于存储池中的元数据卷`beegfs\_m1\_m2\_m5\_m6`、如果使用1.92 TB或3.84 TB驱动器、请将此值设置为`21.25`；对于7.65 TB驱动器、请将此值设置为`22.25`；对于15.3 TB驱动器、请将此值设置为`23.75`。

文件名	Port	浮动IP	NUMA区域	块节点	存储池	所属控制器
meta_01.yml	8015	i1b : 100.127.10 1.1/16 i2b : 100.127.10 2.1/16	0	netapp_01	beegfs_m1_ m2_m3_m3_ m6	答
meta_02.yml	8025	i2b : 100.127.10 2.2/16 i1b : 100.127.10 1.2/ 16	0	netapp_01	beegfs_m1_ m2_m3_m3_ m6	B
meta_03.yml	8035	i3b : 100.127.10 1.3/16 i4b : 100.127.10 2.3/ 16	1.	netapp_02	beegfs_m3_ m4_m7_m8	答

文件名	Port	浮动IP	NUMA区域	块节点	存储池	所属控制器
meta_04.yml	8045	i4b : 100.127.10 2.4/16 i3b : 100.127.10 1.4/ 16	1.	netapp_02	beegfs_m3_ m4_m7_m8	B
meta_05.yml	8055	i1b : 100.127.10 1.5/16 i2b : 100.127.10 2.5/ 16	0	netapp_01	beegfs_m1_ m2_m3_m3_ m6	答
meta_06.yml	8065	i2b : 100.127.10 2.6/16 i1b : 100.127.10 1.6/ 16	0	netapp_01	beegfs_m1_ m2_m3_m3_ m6	B
meta_07.yml	8075	i3b : 100.127.10 1.7/16 i4b : 100.127.10 2.7/ 16	1.	netapp_02	beegfs_m3_ m4_m7_m8	答
meta_08.yml	8085	i4b : 100.127.10 2.8/16 i3b : 100.127.10 1.8/ 16	1.	netapp_02	beegfs_m3_ m4_m7_m8	B

4. 在`group\_vars/`下、使用以下模板为资源组`stor\_01`到`stor\_08`创建文件、然后参考以下示例填写每个服务的占位值：

```
# stor_0X - BeeGFS HA Storage Resource
Groupbeegfs_ha_beegfs_storage_conf_resource_group_options:
  connStoragePortTCP: <PORT>
  connStoragePortUDP: <PORT>
  tuneBindToNumaZone: <NUMA_ZONE>
floating_ips:
  - <PREFERRED PORT:IP/SUBNET>
  - <SECONDARY PORT:IP/SUBNET>
beegfs_service: storage
beegfs_targets:
  <BLOCK NODE>:
    eseries_storage_pool_configuration:
      - name: <STORAGE POOL>
        raid_level: raid6
        criteria_drive_count: 10
        common_volume_configuration:
          segment_size_kb: 512          volumes:
            - size: 21.50 # See note below!          owning_controller:
<OWNING CONTROLLER>
            - size: 21.50          owning_controller: <OWNING
CONTROLLER>
```



有关要使用的正确大小、请参见 ["建议的存储池过度配置百分比"](#)。

文件名	Port	浮动IP	NUMA区域	块节点	存储池	所属控制器
STOR_01.yml	8013	i1b : 100.127.10 3.1/16 i2b : 100.127.10 4.1/ 16	0	netapp_01	beegfs_s1_s2	答
STOR_02.yml	8023	i2b : 100.127.10 4.2/16 i1b : 100.127.10 3.2/ 16	0	netapp_01	beegfs_s1_s2	B
STOR_03.yml	8033	i3b : 100.127.10 3.3/16 i4b : 100.127.10 4.3/ 16	1.	netapp_02	beegfs_s3_s4 .	答
STOR_04.yml	8043	i4b : 100.127.10 4.4/16 i3b : 100.127.10 3.4/ 16	1.	netapp_02	beegfs_s3_s4 .	B

文件名	Port	浮动IP	NUMA区域	块节点	存储池	所属控制器
STOR_05.yml	8053	i1b : 100.127.10 3.5/16 i2b : 100.127.10 4.5/ 16	0	netapp_01	beegfs_s5_s6	答
STOR_06.yml	8063	i2b : 100.127.10 4.6/16 i1b : 100.127.10 3.6/ 16	0	netapp_01	beegfs_s5_s6	B
STOR_07.yml	8073.	i3b : 100.127.10 3.7/16 i4b : 100.127.10 4.7/ 16	1.	netapp_02	beegfs_s7_s8 .	答
STOR_08.yml	8083.	i4b : 100.127.10 4.8/16 i3b : 100.127.10 3.8/ 16	1.	netapp_02	beegfs_s7_s8 .	B

第3步：为元数据+存储构建块配置清单

以下步骤介绍如何为BeeGFS元数据+存储构建块设置Ansible清单。

步骤

1. 在`inventory.yml`中、在现有配置下填充以下参数：

```

meta_09:
  hosts:
    beegfs_03:
    beegfs_04:
stor_09:
  hosts:
    beegfs_03:
    beegfs_04:
meta_10:
  hosts:
    beegfs_03:
    beegfs_04:
stor_10:
  hosts:
    beegfs_03:
    beegfs_04:
meta_11:

```

```
    hosts:
        beegfs_03:
        beegfs_04:
stor_11:
    hosts:
        beegfs_03:
        beegfs_04:
meta_12:
    hosts:
        beegfs_03:
        beegfs_04:
stor_12:
    hosts:
        beegfs_03:
        beegfs_04:
meta_13:
    hosts:
        beegfs_04:
        beegfs_03:
stor_13:
    hosts:
        beegfs_04:
        beegfs_03:
meta_14:
    hosts:
        beegfs_04:
        beegfs_03:
stor_14:
    hosts:
        beegfs_04:
        beegfs_03:
meta_15:
    hosts:
        beegfs_04:
        beegfs_03:
stor_15:
    hosts:
        beegfs_04:
        beegfs_03:
meta_16:
    hosts:
        beegfs_04:
        beegfs_03:
stor_16:
    hosts:
        beegfs_04:
```

2. 在`group\_vars/`下、使用以下模板为资源组`mETA\_09`到`mETA\_16`创建文件、然后参考以下示例填写每个服务的占位值:

```
# meta_0X - BeeGFS HA Metadata Resource Group
beegfs_ha_beegfs_meta_conf_resource_group_options:
  connMetaPortTCP: <PORT>
  connMetaPortUDP: <PORT>
  tuneBindToNumaZone: <NUMA_ZONE>
floating_ips:
  - <PREFERRED PORT:IP/SUBNET>
  - <SECONDARY PORT:IP/SUBNET>
beegfs_service: metadata
beegfs_targets:
  <BLOCK_NODE>:
    eseries_storage_pool_configuration:
      - name: <STORAGE POOL>
        raid_level: raid1
        criteria_drive_count: 4
        common_volume_configuration:
          segment_size_kb: 128
        volumes:
          - size: 21.5 # SEE NOTE BELOW!
            owning_controller: <OWNING CONTROLLER>
```



有关要使用的正确大小、请参见 ["建议的存储池过度配置百分比"](#)。

文件名	Port	浮动IP	NUMA区域	块节点	存储池	所属控制器
meta_09.yml	8015	i1b : 100.127.10 1.9/16 i2b : 100.127.10 2.9/ 16	0	netapp_03	Beegfs_M9_ M10_M13_M 14	答
meta_10.yml	8025	i2b : 100.127.10 2.10/16 i1b : 100.127.10 1.10/ 16	0	netapp_03	Beegfs_M9_ M10_M13_M 14	B
meta_11.yml	8035	i3b : 100.127.2.1 .1/16 i4b : 100.127.10 2.11/16	1.	netapp_04	Beegfs_M11_ M12_M15_16	答

文件名	Port	浮动IP	NUMA区域	块节点	存储池	所属控制器
meta_12.yml	8045	i4b : 100.127.10 2.12/16 i3b : 100.127.10 1.12/ 16	1.	netapp_04	Beegfs_M11_ M12_M15_16	B
meta_13.yml	8055	i1b : 100.127.10 7.13/16 i2b : 100.127.10 2.13/16	0	netapp_03	Beegfs_M9_ M10_M13_M 14	答
meta_14.yml	8065	i2b : 100.127.10 2.14/16 i1b : 100.127.10 1.14/ 16	0	netapp_03	Beegfs_M9_ M10_M13_M 14	B
meta_15.yml	8075	i3b : 100.127.10 1.15/16 i4b : 100.127.10 2.15/ 16	1.	netapp_04	Beegfs_M11_ M12_M15_16	答
meta_16.yml	8085	i4b : 100.127.10 2.16/16 i3b : 100.127.10 1.16/ 16	1.	netapp_04	Beegfs_M11_ M12_M15_16	B

3. 在`group\_vars/下、`使用以下模板为资源组`stor\_09`到`stor\_16`创建文件、然后参考以下示例填写每个服务的占位值：

```
# stor_0X - BeeGFS HA Storage Resource Group
beegfs_ha_beegfs_storage_conf_resource_group_options:
  connStoragePortTCP: <PORT>
  connStoragePortUDP: <PORT>
  tuneBindToNumaZone: <NUMA_ZONE>
floating_ips:
  - <PREFERRED PORT:IP/SUBNET>
  - <SECONDARY PORT:IP/SUBNET>
beegfs_service: storage
beegfs_targets:
  <BLOCK_NODE>:
    eseries_storage_pool_configuration:
      - name: <STORAGE_POOL>
        raid_level: raid6
        criteria_drive_count: 10
        common_volume_configuration:
          segment_size_kb: 512          volumes:
            - size: 21.50 # See note below!
              owning_controller: <OWNING_CONTROLLER>
            - size: 21.50          owning_controller: <OWNING_CONTROLLER>
```



要了解正确的尺寸，请参阅["建议的存储池过度配置百分比"](#) ..

文件名	Port	浮动IP	NUMA区域	块节点	存储池	所属控制器
STOR_09.yml	8013	i1b : 100.127.10 3.9/16 i2b : 100.127.10 4.9/ 16	0	netapp_03	beegfs_s9_s1 0	答
STOR_10.yml	8023	i2b : 100.127.10 4.10/16 i1b : 100.127.10 3.10/ 16	0	netapp_03	beegfs_s9_s1 0	B
STOR_11.yml	8033	i3b : 100.127.10 3.11/16 i4b : 100.127.10 4.11/ 16	1.	netapp_04	beegfs_s11_s 12	答
STOR_12.yml	8043	i4b : 100.127.10 4.12/16 i3b : 100.127.10 3.12/ 16	1.	netapp_04	beegfs_s11_s 12	B

文件名	Port	浮动IP	NUMA区域	块节点	存储池	所属控制器
STOR_13.yml	8053	i1b : 100.127.10 3.13/16 i2b : 100.127.10 4.13/ 16	0	netapp_03	beegfs_s13_s 14	答
STOR_14.yml	8063	i2b : 100.127.10 4.14/16 i1b : 100.127.10 3.14/ 16	0	netapp_03	beegfs_s13_s 14	B
STOR_15.yml	8073.	i3b : 100.127.10 3.15/16 i4b : 100.127.10 4.15/ 16	1.	netapp_04	beegfs_s15_s 16	答
STOR_16.yml	8083.	i4b : 100.127.10 4.16/16 i3b : 100.127.10 3.16/ 16	1.	netapp_04	beegfs_s15_s 16	B

第4步：为纯存储组件配置清单

以下步骤介绍如何为BeeGFS纯存储组件设置Ansible清单。设置元数据+存储与仅存储组件的配置之间的主要区别是、省略了所有元数据资源组、并将每个存储池的`Criteria\_drive\_count`从10更改为12。

步骤

1. 在`inventory.yml`中、在现有配置下填充以下参数：

```
# beegfs_05/beegfs_06 HA Pair (storage only building block):
stor_17:
  hosts:
    beegfs_05:
    beegfs_06:
stor_18:
  hosts:
    beegfs_05:
    beegfs_06:
stor_19:
  hosts:
    beegfs_05:
    beegfs_06:
stor_20:
  hosts:
    beegfs_05:
    beegfs_06:
stor_21:
  hosts:
    beegfs_06:
    beegfs_05:
stor_22:
  hosts:
    beegfs_06:
    beegfs_05:
stor_23:
  hosts:
    beegfs_06:
    beegfs_05:
stor_24:
  hosts:
    beegfs_06:
    beegfs_05:
```

2. 在`group\_vars/`下、使用以下模板为资源组`stor\_17`到`stor\_24`创建文件、然后参考以下示例填写每个服务的占位值：

```
# stor_0X - BeeGFS HA Storage Resource Group
beegfs_ha_beegfs_storage_conf_resource_group_options:
  connStoragePortTCP: <PORT>
  connStoragePortUDP: <PORT>
  tuneBindToNumaZone: <NUMA_ZONE>
floating_ips:
  - <PREFERRED PORT:IP/SUBNET>
  - <SECONDARY PORT:IP/SUBNET>
beegfs_service: storage
beegfs_targets:
  <BLOCK_NODE>:
    eseries_storage_pool_configuration:
      - name: <STORAGE_POOL>
        raid_level: raid6
        criteria_drive_count: 12
        common_volume_configuration:
          segment_size_kb: 512
        volumes:
          - size: 21.50 # See note below!
            owning_controller: <OWNING_CONTROLLER>
          - size: 21.50
            owning_controller: <OWNING_CONTROLLER>
```



要了解正确的尺寸，请参阅["建议的存储池过度配置百分比"](#)。

文件名	Port	浮动IP	NUMA区域	块节点	存储池	所属控制器
STOR_17.yml	8013	i1b : 100.127.10 3.17/16 i2b : 100.127.10 4.17/ 16	0	netapp_05	beegfs_s17_s 18	答
STOR_18.yml	8023	i2b : 100.127.10 4.18/16 i1b : 100.127.10 3.18/ 16	0	netapp_05	beegfs_s17_s 18	B
STOR_19.yml	8033	i3b : 100.127.10 3.19/16 i4b : 100.127.10 4.19/ 16	1.	netapp_06	beegfs_s19_s 20	答

文件名	Port	浮动IP	NUMA区域	块节点	存储池	所属控制器
STOR_20.yml	8043	i4b : 100.127.10 4.20/16 i3b : 100.127.10 3.20/ 16	1.	netapp_06	beegfs_s19_s 20	B
STOR_21.yml	8053	i1b : 100.127.10 3.21/16 i2b : 100.127.10 4.21/ 16	0	netapp_05	Beegfs_S21_ S22	答
STOR_22.yml	8063	i2b : 100.127.10 4.22/16 i1b : 100.127.10 3.22/ 16	0	netapp_05	Beegfs_S21_ S22	B
STOR_23.yml	8073.	i3b : 100.127.10 3.23/16 i4b : 100.127.10 4.23/ 16	1.	netapp_06	beegfs_s23_s 24	答
STOR_24.yml	8083.	i4b : 100.127.10 4.24/16 i3b : 100.127.10 3.24/ 16	1.	netapp_06	beegfs_s23_s 24	B

部署BeeGFS

部署和管理配置涉及运行一个或多个攻略手册、其中包含Ansible需要执行的任务、并将整个系统置于所需状态。

虽然所有任务都可以包含在一本攻略手册中、但对于复杂的系统、这很快就会变得难以管理。Ansible可用于创建和分发角色、以便打包可重复使用的攻略手册和相关内容(例如：默认变量、任务和处理程序)。有关详细信息、请参见Ansible文档 ["角色"](#)。

角色通常作为包含相关角色和模块的Ansible集合的一部分进行分布。因此、这些攻略手册主要仅导入分布在各种NetApp E系列Ansible资料集中的多个角色。



目前、部署BeeGFS至少需要两个组件(四个文件节点)、除非将单独的仲裁设备配置为Tiebreaker、以缓解在双节点集群中建立仲裁时出现的任何问题。

步骤

1. 创建新的`playbook.yml`文件并包括以下内容：

```
# BeeGFS HA (High Availability) cluster playbook.
- hosts: eseries_storage_systems
```

```

gather_facts: false
collections:
  - netapp_eseries.santricity
tasks:
  - name: Configure NetApp E-Series block nodes.
    import_role:
      name: nar_santricity_management
- hosts: all
  any_errors_fatal: true
  gather_facts: false
  collections:
    - netapp_eseries.beegfs
  pre_tasks:
    - name: Ensure a supported version of Python is available on all
      file nodes.
      block:
        - name: Check if python is installed.
          failed_when: false
          changed_when: false
          raw: python --version
          register: python_version
        - name: Check if python3 is installed.
          raw: python3 --version
          failed_when: false
          changed_when: false
          register: python3_version
          when: 'python_version["rc"] != 0 or (python_version["stdout"]
| regex_replace("Python ", "")) is not version("3.0", ">=")'
        - name: Install python3 if needed.
          raw: |
            id=$(grep "^ID=" /etc/*release* | cut -d= -f 2 | tr -d '"')
            case $id in
              ubuntu) sudo apt install python3 ;;
              rhel|centos) sudo yum -y install python3 ;;
              sles) sudo zypper install python3 ;;
            esac
          args:
            executable: /bin/bash
            register: python3_install
            when: python_version['rc'] != 0 and python3_version['rc'] != 0
            become: true
        - name: Create a symbolic link to python from python3.
          raw: ln -s /usr/bin/python3 /usr/bin/python
          become: true
          when: python_version['rc'] != 0
      when: inventory_hostname not in

```

```

groups[beegfs_ha_ansible_storage_group]
  - name: Verify any provided tags are supported.
    fail:
      msg: "{{ item }}" tag is not a supported BeeGFS HA tag. Rerun
your playbook command with --list-tags to see all valid playbook tags."
      when: 'item not in ["all", "storage", "beegfs_ha",
"beegfs_ha_package", "beegfs_ha_configure",
"beegfs_ha_configure_resource", "beegfs_ha_performance_tuning",
"beegfs_ha_backup", "beegfs_ha_client"]'
      loop: "{{ ansible_run_tags }}"
    tasks:
      - name: Verify before proceeding.
        pause:
          prompt: "Are you ready to proceed with running the BeeGFS HA
role? Depending on the size of the deployment and network performance
between the Ansible control node and BeeGFS file and block nodes this
can take awhile (10+ minutes) to complete."
      - name: Verify the BeeGFS HA cluster is properly deployed.
        ansible.builtin.import_role:
          name: netapp_eseries.beegfs.beegfs_ha_7_4

```



本攻略手册将运行几个`pre\_tasks`、用于验证Python 3是否已安装在文件节点上、并检查所提供的Ansible标记是否受支持。

2. 准备好部署BeeGFS时、请使用`Ansible攻略手册`命令和清单和攻略手册文件。

此部署将运行所有`pre\_tasks`、然后提示用户确认、然后再继续实际的BeeGFS部署。

运行以下命令、根据需要调整货叉数量(请参见以下注释):

```
ansible-playbook -i inventory.yml playbook.yml --forks 20
```



尤其是对于大型部署、`forks`建议使用参数覆盖默认的分叉数(5)、以增加Ans可并行配置的主机数量。(有关详细信息, 请参阅["控制攻略手册执行"](#)。)最大值设置取决于可变控制节点上可用的处理功率。以上示例20是在具有4个CPU (Intel® Xeon® Gold 6146 CPU @ 3.20GHz)的虚拟Ansible控制节点上运行的。

根据Ansible控制节点与BeeGFS文件和块节点之间的部署规模和网络性能、部署时间可能会有所不同。

配置BeeGFS客户端

您必须在需要访问BeeGFS文件系统的任何主机上安装和配置BeeGFS客户端、例如计算或GPU节点。在此任务中、您可以使用Ansible和BeeGFS收集。

步骤

1. 如果需要、请从Ansible控制节点向要配置为BeeGFS客户端的每个主机设置无密码SSH:

```
ssh-copy-id <user>@<HOSTNAME_OR_IP>
```

2. 在`host\_vars/`下、为名为`<HOSTNAME>.yaml`的每个BeeGFS客户端创建一个包含以下内容的文件、并使用适用于您环境的正确信息填写占位符文本:

```
# BeeGFS Client
ansible_host: <MANAGEMENT_IP>
# OPTIONAL: If you want to use the NetApp E-Series Host Collection's
# IPoIB role to configure InfiniBand interfaces for clients to connect to
# BeeGFS file systems:
eseries_ipoib_interfaces:
  - name: <INTERFACE>
    address: <IP>/<SUBNET_MASK> # Example: 100.127.1.1/16
  - name: <INTERFACE>
    address: <IP>/<SUBNET_MASK>
```



如果使用双子网寻址方案进行部署、则必须在每个客户端上配置两个InfiniBand接口、两个存储IPoIB子网中的每一个都配置一个。如果为此处列出的每个BeeGFS服务使用示例子网和建议范围,则客户端应在到范围内配置一个接口,而在到范围内配置另一个接口 100.127.1.0 100.127.99.255 100.128.1.0 100.128.99.255。

3. 创建一个新文件`client\_inventory.yaml`、然后在顶部填充以下参数:

```
# BeeGFS client inventory.
all:
  vars:
    ansible_ssh_user: <USER> # This is the user Ansible should use to
    connect to each client.
    ansible_become_password: <PASSWORD> # This is the password Ansible
    will use for privilege escalation, and requires the ansible_ssh_user be
    root, or have sudo privileges.
    The defaults set by the BeeGFS HA role are based on the testing
    performed as part of this NetApp Verified Architecture and differ from
    the typical BeeGFS client defaults.
```



请勿以纯文本格式存储密码。请改用Ansible Vault (请参见的Ansible文档) "[使用Ansible Vault 加密内容](#)")或在运行攻略手册时使用`-ask-pass-pass`选项。

4. 在`client\_inventory.yaml`文件中、列出应在`beegfs\_clients`组下配置为BeeGFS客户端的所有主机、然后指定构建BeeGFS客户端内核模块所需的任何其他配置。

```

children:
  # Ansible group representing all BeeGFS clients:
  beegfs_clients:
    hosts:
      beegfs_01:
      beegfs_02:
      beegfs_03:
      beegfs_04:
      beegfs_05:
      beegfs_06:
      beegfs_07:
      beegfs_08:
      beegfs_09:
      beegfs_10:
    vars:
      # OPTION 1: If you're using the NVIDIA OFED drivers and they are
      already installed:
        eseries_ib_skip: True # Skip installing inbox drivers when using
the IPoIB role.
        beegfs_client_ofed_enable: True
        beegfs_client_ofed_include_path:
"/usr/src/ofa_kernel/default/include"
      # OPTION 2: If you're using inbox IB/RDMA drivers and they are
      already installed:
        eseries_ib_skip: True # Skip installing inbox drivers when using
the IPoIB role.
      # OPTION 3: If you want to use inbox IB/RDMA drivers and need
      them installed/configured.
        eseries_ib_skip: False # Default value.
        beegfs_client_ofed_enable: False # Default value.

```



使用NVIDIA OFED驱动程序时、请确保 `beegfs_client_ofed_include_path` 指向适用于您的Linux安装的正确"headerinclude path"。有关详细信息，请参见BeeGFS文档["RDMA支持"](#)。

5. 在`client\_inventory.yml`文件中、列出要挂载到任何先前定义的`vars`底部的BeeGFS文件系统。

```

    beegfs_client_mounts:
      - sysMgmtHost: 100.127.101.0 # Primary IP of the BeeGFS
management service.
        mount_point: /mnt/beegfs      # Path to mount BeeGFS on the
client.
    connInterfaces:
      - <INTERFACE> # Example: ibs4f1
      - <INTERFACE>
    beegfs_client_config:
      # Maximum number of simultaneous connections to the same
node.

      connMaxInternodeNum: 128 # BeeGFS Client Default: 12
      # Allocates the number of buffers for transferring IO.
      connRDMABufNum: 36 # BeeGFS Client Default: 70
      # Size of each allocated RDMA buffer
      connRDMABufSize: 65536 # BeeGFS Client Default: 8192
      # Required when using the BeeGFS client with the shared-
disk HA solution.
      # This does require BeeGFS targets be mounted in the
default "sync" mode.
      # See the documentation included with the BeeGFS client
role for full details.
      sysSessionChecksEnabled: false

```



beegfs_client_config`表示已测试的设置。有关所有选项的全面概述、请参见`netapp_eseries.beegfs`集合的`beegfs_client`role附带的文档。其中包括有关挂载多个BeeGFS文件系统或多次挂载相同BeeGFS文件系统的详细信息。

6. 创建新的`client\_playbook.yml`文件、然后填充以下参数：

```

# BeeGFS client playbook.
- hosts: beegfs_clients
  any_errors_fatal: true
  gather_facts: true
  collections:
    - netapp_eseries.beegfs
    - netapp_eseries.host
  tasks:
    - name: Ensure IPoIB is configured
      import_role:
        name: ipoib
    - name: Verify the BeeGFS clients are configured.
      import_role:
        name: beegfs_client

```



如果您已在相应的IPoIB接口上安装了所需的IB/RDMA驱动程序和配置的IP、请省略导入`NetApp\_E系列.host`集合和`IPoIB`角色。

7. 要安装和构建客户端并挂载BeeGFS、请运行以下命令：

```
ansible-playbook -i client_inventory.yml client_playbook.yml
```

8. 在将BeeGFS文件系统投入生产之前、我们*强烈*建议您登录到任何客户端并运行`beegfs-fsck -checkfs`以确保所有节点均可访问且未报告任何问题。

扩展到五个组件以上

您可以将Pacemaker和Corosync配置为扩展到五个组件(10个文件节点)以上。但是、较大的集群存在一些缺点、最终、Pacemaker和Corosync最多会占用32个节点。

NetApp仅测试了最多10个节点的BeeGFS HA集群；不建议或不支持将单个集群扩展到超过此限制的范围。但是、BeeGFS文件系统仍需要扩展到10个以上的节点、而NetApp已在NetApp解决方案上的BeeGFS中对此进行了说明。

通过在每个文件系统中部署包含一个子块的多个HA集群、您可以扩展整个BeeGFS文件系统、而不受底层HA集群机制的任何建议限制或硬限制。在此情景中、请执行以下操作：

- 创建一个新的Ansible清单以表示其他HA集群、然后省略配置另一个管理服务。请将每个附加集群`ha\_cluster.yml`中的`beegfs\_ha\_mgmt\_d\_floating\_ip`变量指向第一个BeeGFS管理服务的IP。
- 在向同一文件系统添加其他HA集群时、请确保满足以下要求：
 - BeeGFS节点ID是唯一的。
 - 与`group\_vars`下的每个服务对应的文件名在所有集群中都是唯一的。
 - BeeGFS客户端和服务器IP地址在所有集群中都是唯一的。
 - 在尝试部署或更新其他集群之前、包含BeeGFS管理服务的第一个HA集群正在运行。
- 在每个HA集群各自的目录树中单独维护清单。

尝试在一个目录树中混用多个集群的清单文件可能会导致发生原因 对BeeGFS HA角色聚合应用于特定集群的配置的方式产生问题。



无需在创建新的HA集群之前将每个HA集群扩展到五个组件。在许多情况下、每个集群使用较少的组件会更易于管理。一种方法是将每个机架中的组件配置为HA集群。

建议的存储池过度配置百分比

如果对第二代构建块采用每个存储池配置的标准四个卷、请参见下表。

下表提供了在`eseries\_storage\_pool\_configuration`中为每个BeeGFS元数据或存储目标用作卷大小的建议百分比：

驱动器大小	Size
1.92 TB	18
3.84 TB	21.5
7.68 TB	22.5
15.3 TB	24



上述指南不适用于包含管理服务的存储池、该存储池应将上述大小减少0.25%、以便为管理数据分配1%的存储池。

要了解如何确定这些值、请参见 ["TR-4800：附录A：了解SSD持久性和过度配置"](#)。

大容量组件

标准BeeGFS解决方案 部署指南概述了满足高性能工作负载要求的过程和建议。希望满足大容量要求的客户应观察此处所述的不同部署和建议。



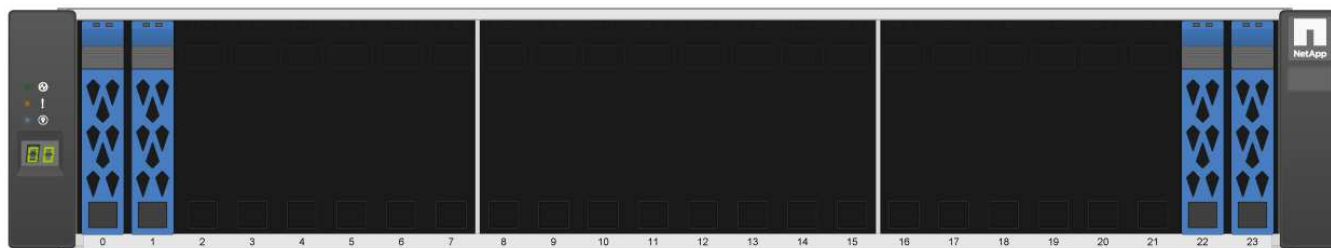
控制器

对于大容量组件、EF600控制器应更换为EF300控制器、每个控制器都安装有一个级联HIC、用于SAS扩展。每个块节点将在阵列机箱中为BeeGFS元数据存储填充最少数量的NVMe SSD、并将其连接到为BeeGFS存储卷填充NL-SAS HDD的扩展架。

文件节点到块节点的配置保持不变。

驱动器放置

每个块节点至少需要4个NVMe SSD才能存储BeeGFS元数据。这些驱动器应放置在机箱最外侧的插槽中。



RAID 1 (2+2) Metadata

扩展托盘

大容量组件的大小可以为每个存储阵列配备1到7个60个驱动器扩展托盘。

有关为每个扩展托盘布线的说明、["有关驱动器架的信息、请参见EF300布线"](#)。

版权信息

版权所有 © 2026 NetApp, Inc.。保留所有权利。中国印刷。未经版权所有者事先书面许可，本文档中受版权保护的任何部分不得以任何形式或通过任何手段（图片、电子或机械方式，包括影印、录音、录像或存储在电子检索系统中）进行复制。

从受版权保护的 NetApp 资料派生的软件受以下许可和免责声明的约束：

本软件由 NetApp 按“原样”提供，不含任何明示或暗示担保，包括但不限于适销性以及针对特定用途的适用性的隐含担保，特此声明不承担任何责任。在任何情况下，对于因使用本软件而以任何方式造成的任何直接性、间接性、偶然性、特殊性、惩罚性或后果性损失（包括但不限于购买替代商品或服务；使用、数据或利润方面的损失；或者业务中断），无论原因如何以及基于何种责任理论，无论出于合同、严格责任或侵权行为（包括疏忽或其他行为），NetApp 均不承担责任，即使已被告知存在上述损失的可能性。

NetApp 保留在不另行通知的情况下随时对本文档所述的任何产品进行更改的权利。除非 NetApp 以书面形式明确同意，否则 NetApp 不承担因使用本文档所述产品而产生的任何责任或义务。使用或购买本产品不表示获得 NetApp 的任何专利权、商标权或任何其他知识产权许可。

本手册中描述的产品可能受一项或多项美国专利、外国专利或正在申请的专利的保护。

有限权利说明：政府使用、复制或公开本文档受 DFARS 252.227-7013（2014 年 2 月）和 FAR 52.227-19（2007 年 12 月）中“技术数据权利 — 非商用”条款第 (b)(3) 条规定的限制条件的约束。

本文档中所含数据与商业产品和/或商业服务（定义见 FAR 2.101）相关，属于 NetApp, Inc. 的专有信息。根据本协议提供的所有 NetApp 技术数据和计算机软件具有商业性质，并完全由私人出资开发。美国政府对这些数据的使用权具有非排他性、全球性、受限且不可撤销的许可，该许可既不可转让，也不可再许可，但仅限在与交付数据所依据的美国政府合同有关且受合同支持的情况下使用。除本文档规定的情形外，未经 NetApp, Inc. 事先书面批准，不得使用、披露、复制、修改、操作或显示这些数据。美国政府对国防部的授权仅限于 DFARS 的第 252.227-7015(b)（2014 年 2 月）条款中明确的权利。

商标信息

NetApp、NetApp 标识和 <http://www.netapp.com/TM> 上所列的商标是 NetApp, Inc. 的商标。其他公司和产品名称可能是其各自所有者的商标。