



Utilice arquitecturas verificadas

BeeGFS on NetApp with E-Series Storage

NetApp
January 27, 2026

Tabla de contenidos

Utilice arquitecturas verificadas	1
Descripción general y requisitos	1
Descripción general de la solución	1
Información general de la arquitectura	2
Requisitos técnicos	6
Revisar el diseño de la solución	9
Descripción general del diseño	10
Configuración de hardware	10
Configuración de software	12
Verificación del diseño	19
Directrices de tamaño	25
Ajuste del rendimiento	26
Elemento básico de gran capacidad	28
Ponga en marcha la solución	29
Información general sobre la implementación	29
Obtenga más información sobre el inventario de Ansible	31
Revise las prácticas recomendadas	33
Ponga en marcha el hardware	37
Poner en marcha el software	40
Escalabilidad más allá de cinco elementos básicos	80
Se recomendaron porcentajes de sobreaprovisionamiento del pool de almacenamiento	81
Elemento básico de gran capacidad	81

Utilice arquitecturas verificadas

Descripción general y requisitos

Descripción general de la solución

La solución BeeGFS en NetApp combina el sistema de archivos BeeGFS en paralelo con los sistemas de almacenamiento EF600 de NetApp para obtener una infraestructura fiable, escalable y rentable que pueda seguir el ritmo de las cargas de trabajo más exigentes.

Programa NVA

BeeGFS en la solución de NetApp forma parte del programa Arquitectura verificada de NetApp (NVA), que proporciona a los clientes configuraciones de referencia y directrices para el ajuste de tamaño en cargas de trabajo específicas y casos prácticos. Las soluciones de NVA se han probado exhaustivamente y están diseñadas para minimizar los riesgos de la puesta en marcha y acelerar el plazo de comercialización.

Perspectiva general de diseño

La solución BeeGFS en NetApp está diseñada como una arquitectura de elementos básicos escalable, configurable para varias cargas de trabajo exigentes. Ya sea para tratar muchos archivos pequeños, gestionar operaciones sustanciales de archivos grandes o una carga de trabajo híbrida, el sistema de archivos se puede personalizar para satisfacer estas necesidades. La alta disponibilidad está integrada en el diseño con el uso de una estructura de hardware de dos niveles que permite la recuperación tras fallos independiente en varias capas de hardware y garantiza un rendimiento constante, incluso durante degradaciones parciales del sistema. El sistema de archivos BeeGFS permite un entorno escalable y de alto rendimiento entre diferentes distribuciones de Linux y ofrece a los clientes un único espacio de nombres de almacenamiento de fácil acceso. Obtenga más información en el "[información general sobre la arquitectura](#)".

Casos de uso

Los siguientes casos prácticos se aplican a BeeGFS en la solución de NetApp:

- Sistemas DGX SuperPOD de NVIDIA que incorporan DGX con GPU A100, H100, H200 y B200.
- Inteligencia artificial (IA), incluido el aprendizaje automático (ML), el aprendizaje profundo (DL), el procesamiento de lenguaje natural a gran escala (NLP) y la comprensión del lenguaje natural (NLU). Para obtener más información, consulte "[BeeGFS para IA: Realidad versus ficción](#)".
- Informática de alto rendimiento (HPC), incluidas aplicaciones aceleradas por MPI (interfaz de paso de mensajes) y otras técnicas informáticas distribuidas. Para obtener más información, consulte "[Por qué BeeGFS va más allá del HPC](#)".
- Cargas de trabajo de aplicaciones caracterizadas por:
 - Leer o escribir en archivos de más de 1 GB
 - Leyendo o escribiendo en el mismo archivo por varios clientes (10s, 100s y 1000s).
- Conjuntos de datos de varios terabytes o varios petabytes.
- Entornos que requieren un único espacio de nombres de almacenamiento optimizable para una combinación de archivos grandes y pequeños.

Beneficios

Entre las ventajas clave del uso de BeeGFS en NetApp se incluyen:

- Disponibilidad de diseños de hardware verificados que proporcionan una integración completa de los componentes de hardware y software para garantizar un rendimiento y una fiabilidad previsibles.
- Puesta en marcha y gestión con Ansible para obtener más simplicidad y coherencia a escala.
- Supervisión y observabilidad proporcionadas mediante el Analizador de rendimiento de E-Series y el complemento BeeGFS. Para obtener más información, consulte "[Presentación de un marco para supervisar las soluciones E-Series de NetApp](#)".
- Alta disponibilidad con una arquitectura de discos compartidos que proporciona durabilidad y disponibilidad de los datos.
- Compatibilidad con la gestión y orquestación de cargas de trabajo modernas mediante contenedores y Kubernetes. Para obtener más información, consulte "[Kubernetes se ha encontrado con BeeGFS: Un ejemplo de inversión lista para el futuro](#)".

Información general de la arquitectura

BeeGFS en la solución de NetApp incluye consideraciones de diseño arquitectónico utilizadas para determinar el equipo, el cableado y las configuraciones específicos necesarios para admitir cargas de trabajo validadas.

Arquitectura de elementos básicos

El sistema de archivos BeeGFS se puede implementar y escalar de diferentes maneras en función de los requisitos de almacenamiento. Por ejemplo, los casos de uso que incluyan principalmente numerosos ficheros pequeños se beneficiarán de un rendimiento y una capacidad adicionales relacionados con los metadatos, mientras que los casos de uso que incluyan menos archivos de gran tamaño pueden favorecer una mayor capacidad de almacenamiento y un mayor rendimiento en el contenido real de los ficheros. Estas múltiples consideraciones afectan a las diferentes dimensiones de la implementación del sistema de archivos paralelos, lo que añade complejidad al diseño y la implementación del sistema de archivos.

Para hacer frente a estos retos, NetApp ha diseñado una arquitectura de elementos básicos estándar que se utiliza para escalar horizontalmente cada una de estas dimensiones. Normalmente, los bloques de creación de BeeGFS se implementan en uno de los tres perfiles de configuración:

- Un único elemento básico, incluidos los servicios de gestión, metadatos y almacenamiento de BeeGFS
- Metadatos BeeGFS más un elemento básico de almacenamiento
- Un elemento básico de sólo almacenamiento BeeGFS

El único cambio de hardware entre estas tres opciones es el uso de unidades más pequeñas para los metadatos de BeeGFS. De lo contrario, todos los cambios de configuración se aplican a través del software. Con Ansible como motor de puesta en marcha, configurar el perfil deseado para un elemento básico concreto supone que las tareas de configuración sean sencillas.

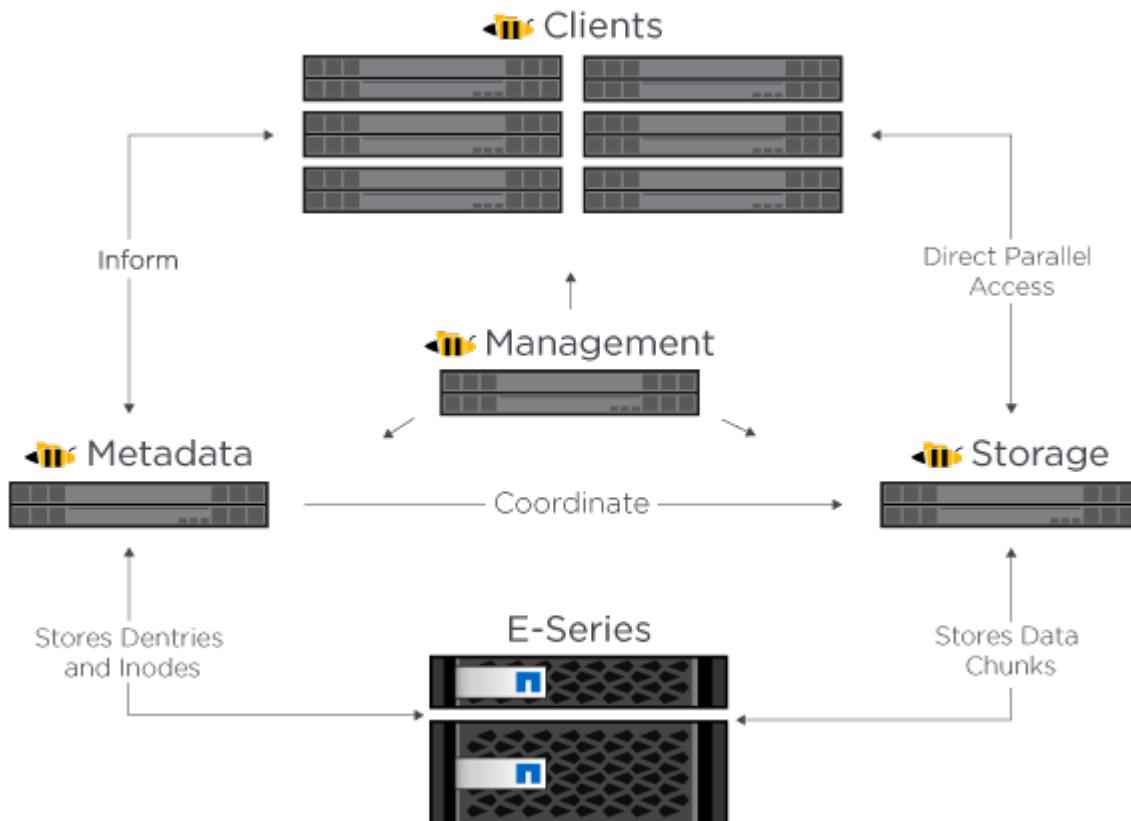
Para obtener información detallada, consulte [Diseño de hardware verificado](#).

Servicios de ficheros

El sistema de archivos BeeGFS incluye los siguientes servicios principales:

- **Servicio de administración.** registra y supervisa todos los demás servicios.
- **Servicio de almacenamiento.** almacena el contenido del archivo de usuario distribuido conocido como archivos de fragmentos de datos.
- **Servicio de metadatos.** realiza un seguimiento del diseño del sistema de archivos, del directorio, de los atributos del archivo, etc.
- **Servicio de cliente.** se cuenta con el sistema de archivos para acceder a los datos almacenados.

En la siguiente figura, se muestran los componentes de la solución BeeGFS y las relaciones que se utilizan con los sistemas E-Series de NetApp.



Como sistema de archivos paralelo, BeeGFS segmenta sus archivos en varios nodos de servidor para maximizar el rendimiento de lectura/escritura y la escalabilidad. Los nodos de servidor funcionan juntos para proporcionar un único sistema de archivos que se puede montar y acceder simultáneamente por otros nodos de servidor, comúnmente conocidos como *clients*. Estos clientes pueden ver y consumir el sistema de archivos distribuido de forma similar a un sistema de archivos local como NTFS, XFS o ext4.

Los cuatro servicios principales se ejecutan en una amplia gama de distribuciones de Linux compatibles y se comunican a través de cualquier red compatible con TCP/IP o RDMA, incluidas InfiniBand (IB), Omni-Path (OPA) y RDMA over Converged Ethernet (roce). Los servicios de servidor BeeGFS (gestión, almacenamiento y metadatos) son daemons de espacio de usuario, mientras que el cliente es un módulo de kernel nativo (sin parches). Todos los componentes se pueden instalar o actualizar sin reiniciar, y se puede ejecutar cualquier combinación de servicios en el mismo nodo.

Arquitectura de ALTA DISPONIBILIDAD

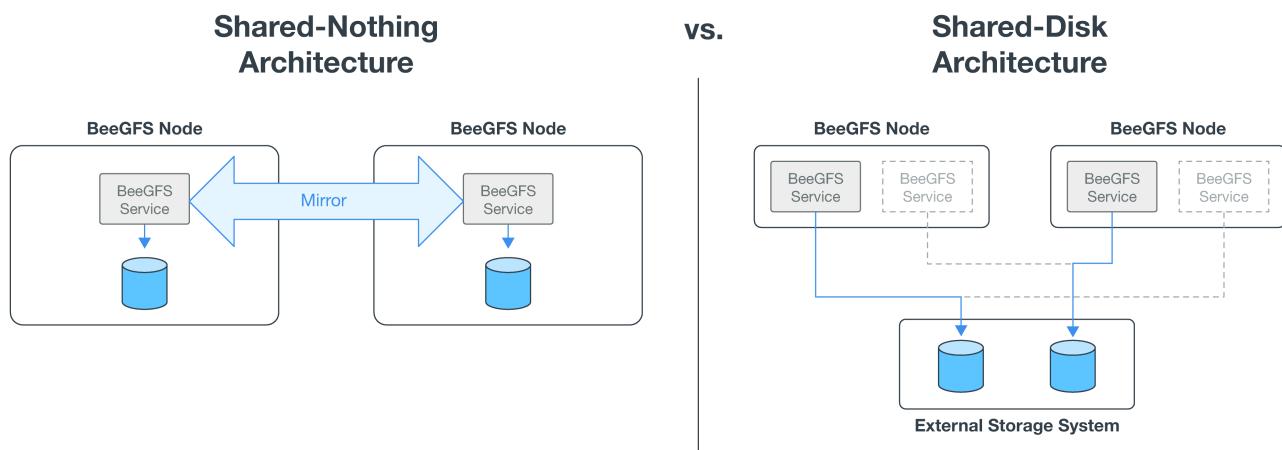
BeeGFS en NetApp amplía la funcionalidad de la edición empresarial de BeeGFS mediante la creación de una solución totalmente integrada en hardware de NetApp que permite una arquitectura de alta disponibilidad (ha)

de disco compartido.



Aunque la edición de la comunidad BeeGFS puede utilizarse de forma gratuita, la edición para empresas requiere adquirir un contrato de suscripción de soporte profesional de un partner como NetApp. La edición empresarial permite utilizar varias funciones adicionales como la resiliencia, la aplicación de cuotas y pools de almacenamiento.

En la figura siguiente se comparan las arquitecturas de alta disponibilidad de disco compartido y nada compartido.



Para obtener más información, consulte ["Anuncio de alta disponibilidad de BeeGFS compatible con NetApp"](#).

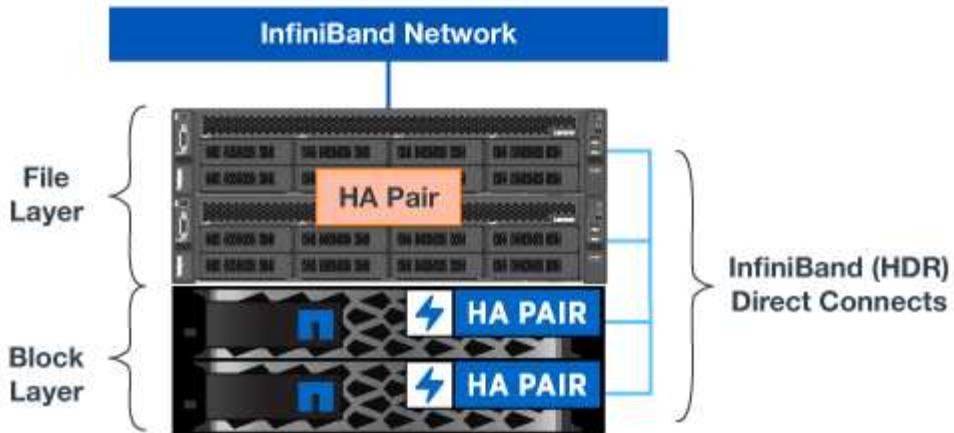
Nodos verificados

La solución BeeGFS en NetApp ha verificado los nodos que se indican a continuación.

Nodo	Hardware subyacente	Detalles
Bloque	Sistema de almacenamiento EF600 de NetApp	Una cabina de almacenamiento 2U íntegramente con NVMe de alto rendimiento diseñada para cargas de trabajo exigentes.
Archivo	Servidor Lenovo ThinkSystem SR665 V3	Un servidor 2U de dos zócalos con PCIe 5,0, procesadores dobles AMD EPYC 9124. Para obtener más información sobre el Lenovo SR665 V3, consulte "El sitio web de Lenovo" .
	Servidor Lenovo ThinkSystem SR665	Un servidor 2U de dos zócalos con PCIe 4,0, procesadores dobles AMD EPYC 7003. Para obtener más información sobre el Lenovo SR665, consulte "El sitio web de Lenovo" .

Diseño de hardware verificado

Los elementos básicos de la solución (que se muestran en la siguiente figura) utilizan los servidores de nodos de archivo verificados para la capa de archivo BeeGFS y dos sistemas de almacenamiento EF600 como capa de bloques.



La solución BeeGFS en NetApp se ejecuta en todos los elementos básicos de la puesta en marcha. El primer elemento básico puesto en marcha debe ejecutar los servicios de gestión, metadatos y almacenamiento de BeeGFS (conocido como el elemento básico). Todos los componentes posteriores se pueden configurar a través de software para ampliar los metadatos y los servicios de almacenamiento, o bien para proporcionar servicios de almacenamiento de forma exclusiva. Este enfoque modular permite escalar el sistema de archivos a las necesidades de una carga de trabajo utilizando las mismas plataformas de hardware subyacentes y el diseño de elementos básicos.

Se pueden desplegar hasta cinco elementos básicos para formar un clúster de alta disponibilidad de Linux independiente. Esto optimiza la gestión de recursos con Pacemaker y mantiene una sincronización eficiente con Corosync. Uno o varios de estos clústeres HA de BeeGFS independientes se han combinado para crear un sistema de archivos BeeGFS al que pueden acceder los clientes como un único espacio de nombres de almacenamiento. En cuanto al hardware, un solo rack de 42U puede acomodar hasta cinco elementos básicos, junto con dos switches InfiniBand de 1U Gb para la red de almacenamiento/datos. Consulte el gráfico siguiente para obtener una representación visual.

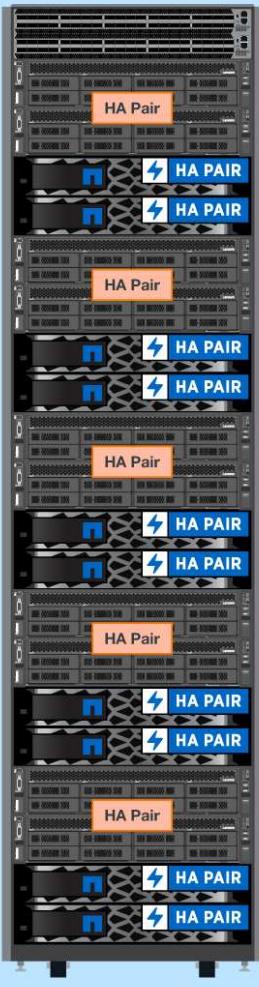


Se necesita un mínimo de dos bloques de construcción para establecer el quórum en el clúster de commutación por error. Un clúster de dos nodos tiene limitaciones que podrían impedir que se produzca una commutación al respaldo correcta. Puede configurar un clúster de dos nodos incorporando un tercer dispositivo como tiebreaker; sin embargo, esta documentación no describe ese diseño.

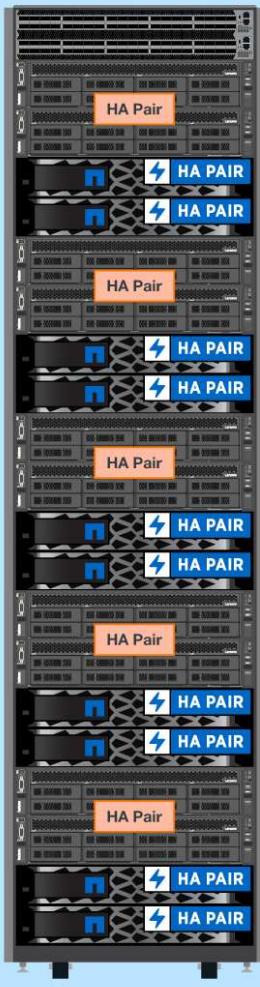


BeeGFS Parallel Filesystem

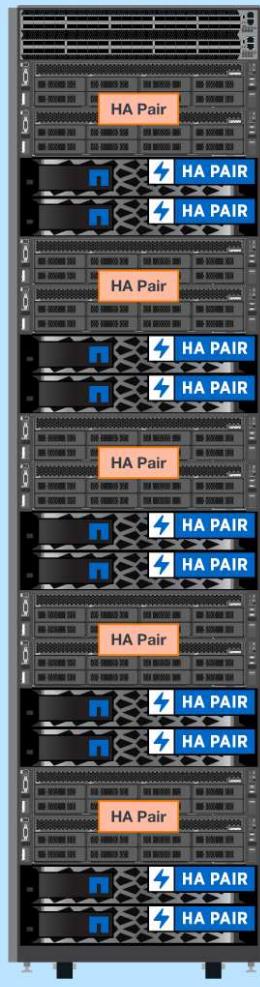
Standalone HA Cluster



Standalone HA Cluster



Standalone HA Cluster



Ansible

BeeGFS en NetApp se entrega y se pone en marcha mediante la automatización de Ansible, que se encuentra alojado en GitHub y Ansible Galaxy (puede acceder a la colección BeeGFS en "[Galaxia de ansible](#)" y.. "[GitHub de E-Series de NetApp](#)"). A pesar de que Ansible se prueba principalmente con el hardware utilizado para ensamblar los elementos básicos de BeeGFS, puede configurarlo para que se ejecute prácticamente en cualquier servidor basado en x86 utilizando una distribución de Linux compatible.

Para obtener más información, consulte "[Puesta en marcha de BeeGFS con almacenamiento E-Series](#)".

Requisitos técnicos

Para implantar BeeGFS en la solución NetApp, asegúrese de que su entorno cumple los requisitos tecnológicos descritos en este documento.

Requisitos de hardware

Antes de empezar, asegúrese de que su hardware cumpla con las siguientes especificaciones para el diseño de elementos básicos de segunda generación de la solución BeeGFS en NetApp. Los componentes exactos para una puesta en marcha concreta pueden variar en función de las necesidades del cliente.

Cantidad	Componente de hardware	Requisitos
2	Nodos de archivo BeeGFS	<p>Cada nodo de archivo debe cumplir o superar las especificaciones de los nodos de archivo recomendados para lograr el rendimiento esperado.</p> <p>Opciones de nodo de archivo recomendadas:</p> <ul style="list-style-type: none"> • Lenovo ThinkSystem SR665 V3 <ul style="list-style-type: none"> ◦ Procesadores: 2x AMD EPYC 9124 16C 3,0 GHz (configurado como dos zonas NUMA). ◦ Memoria: 256GB (16x 16GB TruDDR5 4800MHz RDIMM-A) ◦ Expansión PCIe: Cuatro ranuras PCIe Gen5 x16 (dos por zona NUMA) ◦ Miscelánea: <ul style="list-style-type: none"> ▪ Dos unidades en RAID 1 para sistemas operativos (SATA de 1TB 7,2K TB o superior) ▪ Puerto 1GbE para gestión de SO en banda ▪ 1GbE BMC con API de Redfish para la gestión de servidores fuera de banda ▪ Sistemas de alimentación y ventiladores de rendimiento duales intercambiables en caliente
2	Nodos de bloque de E-Series (cabina EF600)	Memoria: 256GB (128GB por controlador). Adaptador: 2 puertos 200GB/HDR (NVMe/IB). Drives: configurado para que coincida con los metadatos deseados y la capacidad de almacenamiento.
8	Adaptadores de tarjeta de host InfiniBand (para nodos de archivo).	Los adaptadores de tarjeta host pueden variar según el modelo de servidor del nodo de archivo. Las recomendaciones para los nodos de archivo verificados incluyen: <ul style="list-style-type: none"> • Servidor Lenovo ThinkSystem SR665 V3: <ul style="list-style-type: none"> ◦ MCX755106AS-HEAT ConnectX-7, NDR200, QSFP112, 2 puertos, PCIe Gen5 x16, adaptador InfiniBand
1	Switch de red de almacenamiento	El switch de red de almacenamiento debe tener una capacidad de velocidades InfiniBand de 200GB Gb/s. Los modelos de conmutación recomendados incluyen: <ul style="list-style-type: none"> • Interruptor NVIDIA QM9700 Quantum 2 NDR InfiniBand • Interruptor NVIDIA MQM8700 Quantum HDR InfiniBand

Requisitos de cableado

Conexiones directas desde nodos de bloque a nodos de archivo.

Cantidad	Número de pieza	Longitud
8	MCP1650-H001E30 (cable de cobre pasivo NVIDIA, QSFP56, 200GB/s)	1 m

Conexiones desde los nodos de archivos al conmutador de red de almacenamiento. Seleccione la opción de cable adecuada de la siguiente tabla según su switch de almacenamiento InfiniBand. + La longitud del cable recomendada es de 2m m; sin embargo, esto puede variar en función del entorno del cliente.

Modelo de switch	Tipo de cable	Cantidad	Número de pieza
NVIDIA QM9700	Fibra activa (incluidos los transceptores)	2	MMA4Z00-NS (multimodo, IB/ETH, 800GB Gb/s 2x400 Gb/s OSFP de dos puertos)
		4	MFP7E20-Nxxx (multimodo, cable de fibra divisora de 4 canales a dos de 2 canales)
		8	MMA1Z00-NS400 (multimodo, IB/ETH, un puerto QSFP-112 de 400GB Gb/s)
	Cobre pasivo	2	MCP7Y40-N002 (cable divisor de cobre pasivo NVIDIA, InfiniBand 800GB/s a 4x 200GB/s, OSFP a 4x QSFP112)
NVIDIA MQM8700	Fibra activa	8	MFS1S00-H003E (cable de fibra activa NVIDIA, InfiniBand 200GB Gb/s, QSFP56 m)
	Cobre pasivo	8	MCP1650-H002E26 (cable de cobre pasivo NVIDIA, InfiniBand 200GB Gb/s, QSFP56 m)

Requisitos de software y firmware

Para garantizar un rendimiento y una fiabilidad previsibles, las versiones de la solución BeeGFS en NetApp se han probado con versiones específicas de componentes de software y firmware. Estas versiones son necesarias para implementar la solución.

Requisitos del nodo de archivo

De NetApp	Versión
Red Hat Enterprise Linux (RHEL)	Servidor físico RHEL 9.4 con alta disponibilidad (2 sockets). Nota: Los nodos de archivo requieren una suscripción válida a Red Hat Enterprise Linux Server y el complemento de alta disponibilidad de Red Hat Enterprise Linux.
Kernel de Linux	5.14.0-427.42.1.el9_4.x86_64
Firmware de HCA	Firmware ConnectX-7 HCA FW: 28.45.1200 + PXE: 3.7.0500 + UEFI: 14.38.0016 ConnectX-6 HCA Firmware FW: 20.43.2566 + PXE: 3.7.0500 + UEFI: 14.37.0013

Requisitos del nodo de bloques de EF600

De NetApp	Versión
Sistema operativo SANtricity	11.90R3
NVSRAM	N6000-890834-D02.dlp
Firmware de la unidad	La última versión disponible para los modelos de unidad en uso. Consulte la " Sitio del firmware de un disco E-Series ".

Requisitos de puesta en marcha de software

En la siguiente tabla se enumeran los requisitos de software puestos en marcha automáticamente como parte de la puesta en marcha de BeeGFS basada en Ansible.

De NetApp	Versión
BeeGFS	7.4.6
Corosync	3.1.8-1
Marcapasos	2.1.7-5,2
PC	0.11.7-2
Agentes de valla (sepia/apc)	4.10.0-62
Controladores InfiniBand/RDMA	MLNX_OFED_LINUX-23.10-3.2.2.1-LTS

Requisitos del nodo de control de Ansible

BeeGFS en la solución de NetApp se pone en marcha y se gestiona desde un nodo de control de Ansible. Para obtener más información, consulte "[Documentación de Ansible](#)".

Los requisitos de software que se enumeran en las siguientes tablas son específicos de la versión de la colección de Ansible BeeGFS de NetApp que se indica a continuación.

De NetApp	Versión
Ansible	10.x
Núcleo Ansible	>= 2.13.0
Python	3,10
Paquetes de Python adicionales	Criptografía-43.0.0, netaddr-1.3.0, ipaddr-2.2.0
Colección Ansible BeeGFS de NetApp E-Series	3.2.0

Revisar el diseño de la solución

Descripción general del diseño

Se requieren equipos, cables y configuraciones específicos para admitir BeeGFS en la solución de NetApp, que combina el sistema de archivos en paralelo BeeGFS con los sistemas de almacenamiento EF600 de NetApp.

Obtenga más información:

- "[Configuración de hardware](#)"
- "[Configuración de software](#)"
- "[Verificación del diseño](#)"
- "[Directrices de tamaño](#)"
- "[Ajuste del rendimiento](#)"

Arquitecturas derivadas con variaciones en el diseño y el rendimiento:

- "[Elementos básicos de alta capacidad](#)"

Configuración de hardware

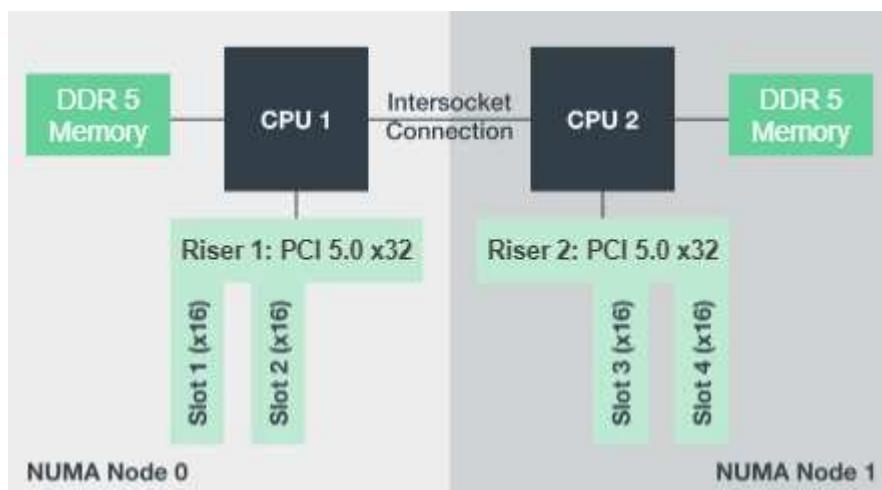
La configuración de hardware para BeeGFS en NetApp incluye nodos de archivo y cableado de red.

Configuración de nodos de archivos

Los nodos de archivos tienen dos sockets de CPU configurados como zonas NUMA independientes, que incluyen acceso local a un mismo número de ranuras PCIe y memoria.

Los adaptadores InfiniBand deben llenarse en las ranuras o elevadores PCI adecuados, por lo que la carga de trabajo se equilibrará entre los canales PCIe y los canales de memoria disponibles. Equilibre la carga de trabajo aislando completamente el trabajo de los servicios BeeGFS individuales a un nodo NUMA en particular. El objetivo es lograr un rendimiento similar en cada nodo de archivo como si se tratara de dos servidores de un único socket independientes.

En la figura siguiente se muestra la configuración NUMA del nodo de archivo.



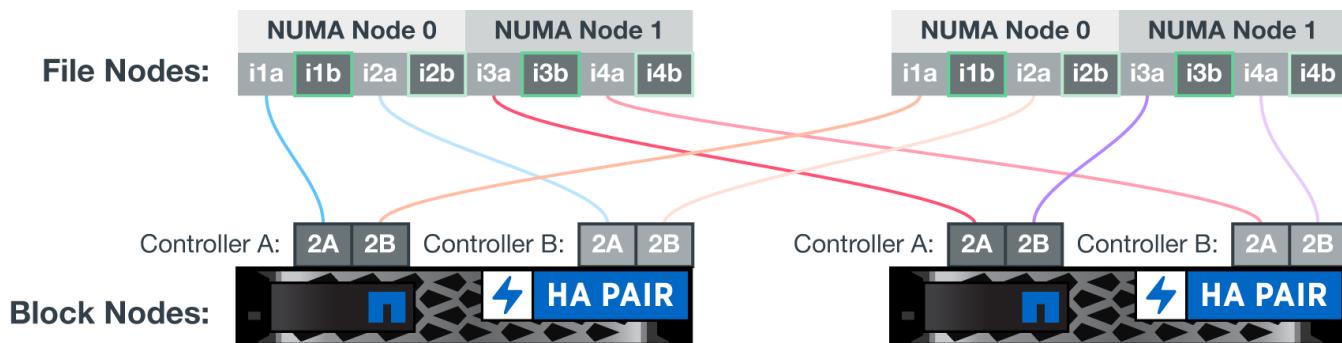
Los procesos BeeGFS se anclan a una zona NUMA en particular para garantizar que las interfaces utilizadas se encuentren en la misma zona. Esta configuración evita la necesidad de acceso remoto a través de la conexión entre sockets. La conexión entre zócalos se conoce a veces como el enlace QPI o GMI2; incluso en arquitecturas de procesador modernas, pueden crear un cuello de botella al utilizar redes de alta velocidad como HDR InfiniBand.

Configuración del cableado de red

Dentro de un elemento básico, cada nodo de archivo está conectado a dos nodos de bloques mediante un total de cuatro conexiones InfiniBand redundantes. Además, cada nodo de archivo tiene cuatro conexiones redundantes a la red de almacenamiento de InfiniBand.

En la siguiente figura, observe que:

- Todos los puertos de nodos de archivos delineados en verde se utilizan para conectarse al entramado de almacenamiento; todos los demás puertos de nodos de archivos son los puertos de nodos de bloques.
- Dos puertos InfiniBand en una zona NUMA específica se conectan a las controladoras A y B del mismo nodo de bloque.
- Los puertos del nodo NUMA 0 siempre se conectan al primer nodo de bloque.
- Los puertos del nodo NUMA 1 se conectan al segundo nodo de bloque.



Cuando se utilizan cables divisorios para conectar el switch de almacenamiento a nodos de archivos, un cable debe ramificarse y conectarse a los puertos descritos en verde claro. Otro cable debe ramificarse y conectarse a los puertos señalados en verde oscuro. Además, para las redes de almacenamiento con switches redundantes, los puertos descritos en verde claro deben conectarse a un switch, mientras que los puertos de verde oscuro deben conectarse a otro switch.

La configuración del cableado que se muestra en la figura permite a cada servicio BeeGFS:

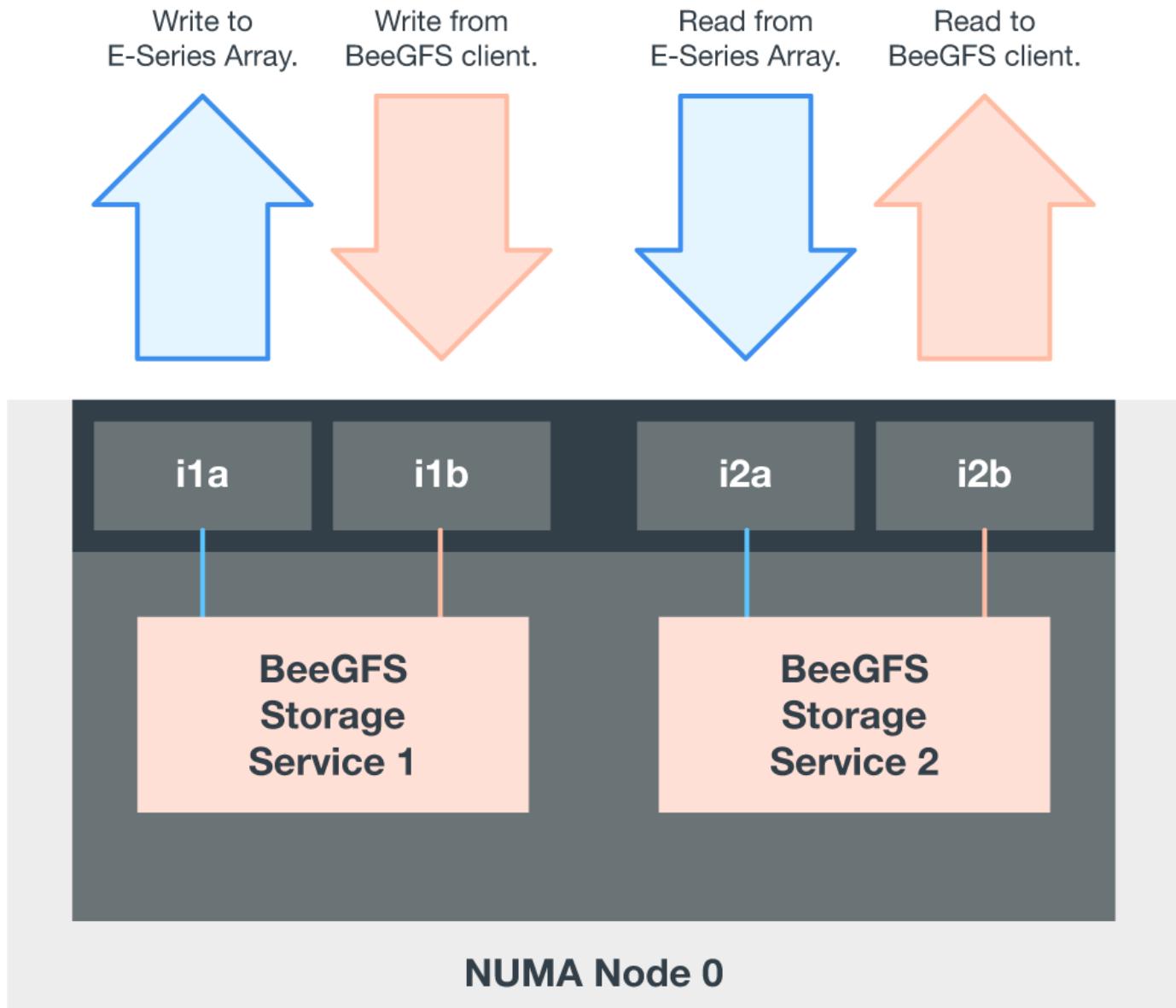
- Se ejecuta en la misma zona NUMA independientemente del nodo de archivo que esté ejecutando el servicio BeeGFS.
- Tener rutas secundarias óptimas a la red de almacenamiento front-end y a los nodos de bloques de back-end, independientemente de dónde se produzca un fallo.
- Minimice los efectos en el rendimiento si un nodo de archivo o una controladora de un nodo de bloque requiere mantenimiento.

Cableado para aprovechar el ancho de banda

Para aprovechar todo el ancho de banda bidireccional de PCIe, asegúrese de que un puerto de cada adaptador InfiniBand se conecta a la estructura de almacenamiento y el otro puerto se conecta a un nodo de

bloque.

La siguiente figura muestra el diseño del cableado utilizado para aprovechar todo el ancho de banda bidireccional de PCIe.



Para cada servicio BeeGFS, utilice el mismo adaptador para conectar el puerto preferido utilizado para el tráfico de cliente con la ruta al controlador de nodos de bloque que es el propietario principal de dichos volúmenes de servicios. Para obtener más información, consulte "[Configuración de software](#)".

Configuración de software

La configuración de software de BeeGFS en NetApp incluye componentes de red BeeGFS, nodos de bloque EF600, nodos de archivos BeeGFS, grupos de recursos y servicios BeeGFS.

Configuración de red BeeGFS

La configuración de red BeeGFS consta de los siguientes componentes.

- **IP flotantes** las IP flotantes son un tipo de dirección IP virtual que se puede enrutar dinámicamente a cualquier servidor de la misma red. Varios servidores pueden tener la misma dirección IP flotante, pero sólo puede estar activa en un servidor en un momento dado.

Cada servicio de servidor BeeGFS tiene su propia dirección IP que puede moverse entre nodos de archivo en función de la ubicación de ejecución del servicio de servidor BeeGFS. Esta configuración de IP flotante permite que cada servicio commute por error de manera independiente al otro nodo de archivo. El cliente simplemente necesita conocer la dirección IP de un servicio BeeGFS concreto; no necesita saber qué nodo de archivo está ejecutando ese servicio en ese momento.

- **Configuración de hosts múltiples del servidor BeeGFS** para aumentar la densidad de la solución, cada nodo de archivo tiene varias interfaces de almacenamiento con IP configuradas en la misma subred IP.

Es necesario configurar más para asegurarse de que esta configuración funciona de la forma esperada con el paquete de redes de Linux, ya que, de forma predeterminada, es posible responder las solicitudes a una interfaz en otra interfaz si sus IP se encuentran en la misma subred. Además de otros inconvenientes, este comportamiento predeterminado hace que sea imposible establecer o mantener correctamente las conexiones RDMA.

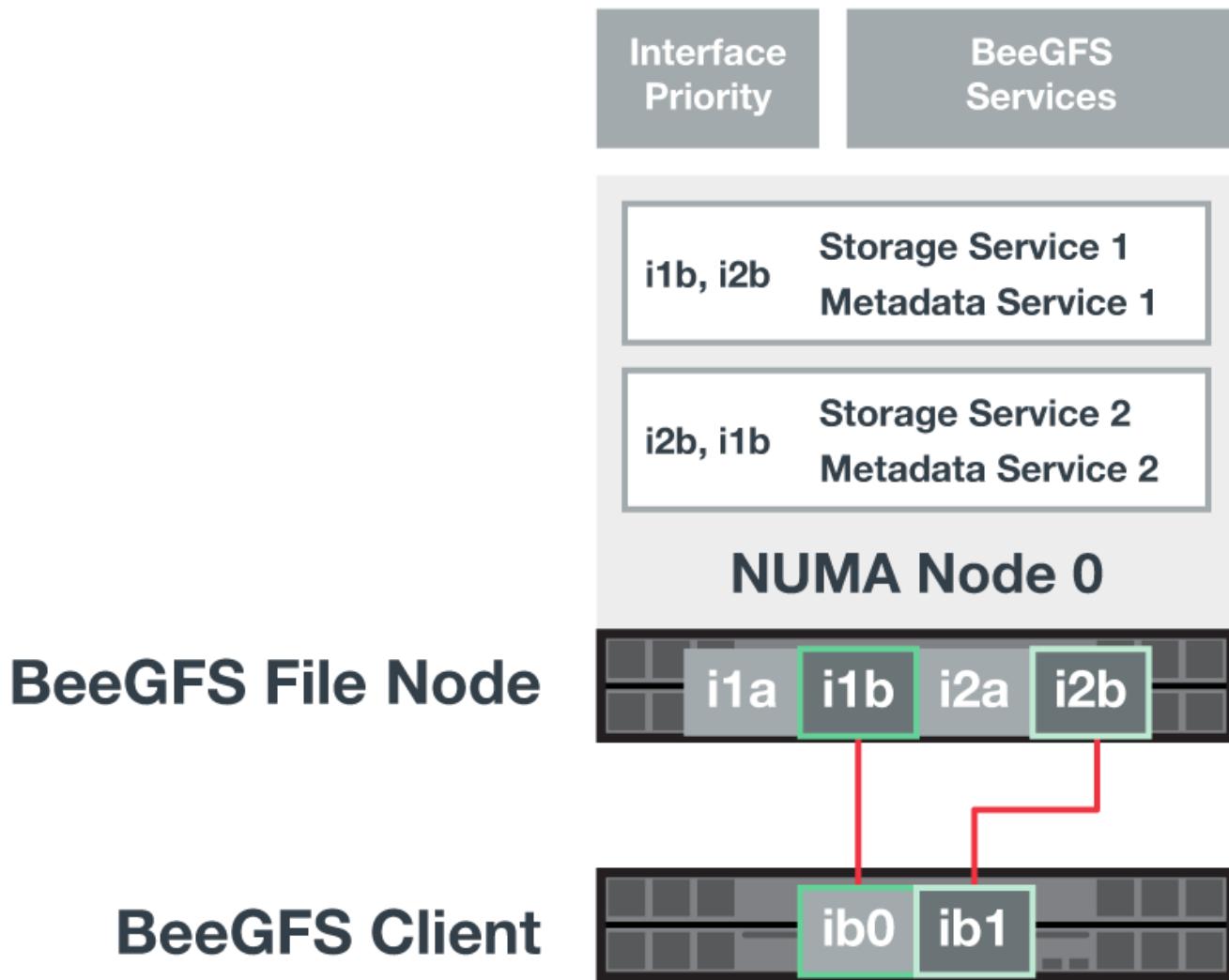
La puesta en marcha basada en Ansible gestiona el apriete del comportamiento de la ruta inversa (RP) y del protocolo de resolución de direcciones (ARP), junto con la garantía de cuándo se inician y se detienen las IP flotantes; las reglas y rutas IP correspondientes se crean de forma dinámica para permitir que la configuración de red de múltiples hosts funcione correctamente.

- **Configuración multicarril del cliente BeeGFS** *Multi-rail* se refiere a la capacidad de una aplicación para utilizar múltiples conexiones de red independientes, o “rieles”, para aumentar el rendimiento.

BeeGFS implementa soporte multi-rail para permitir el uso de múltiples interfaces IB en una sola subred IPoIB. Esta funcionalidad habilita funciones como el equilibrio de carga dinámico entre NIC de RDMA y optimiza el uso de recursos de red. También se integra con el almacenamiento GPUDirect (GDS) de NVIDIA, que ofrece un mayor ancho de banda del sistema y reduce la latencia y el uso de la CPU del cliente.

Esta documentación proporciona instrucciones para configuraciones de subred IPoIB única. Se admiten configuraciones de subred IPoIB dobles, pero no ofrecen las mismas ventajas que las configuraciones de subred única.

En la siguiente figura se muestra el equilibrio del tráfico en varias interfaces de cliente BeeGFS.



Debido a que cada archivo de BeeGFS suele estar segado en múltiples servicios de almacenamiento, la configuración multicanal permite al cliente conseguir un mayor rendimiento del que es posible con un único puerto InfiniBand. Por ejemplo, el siguiente ejemplo de código muestra una configuración común de segmentación de archivos que permite al cliente equilibrar el tráfico entre ambas interfaces:

+

```
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]
```

Configuración de nodos de bloques de EF600

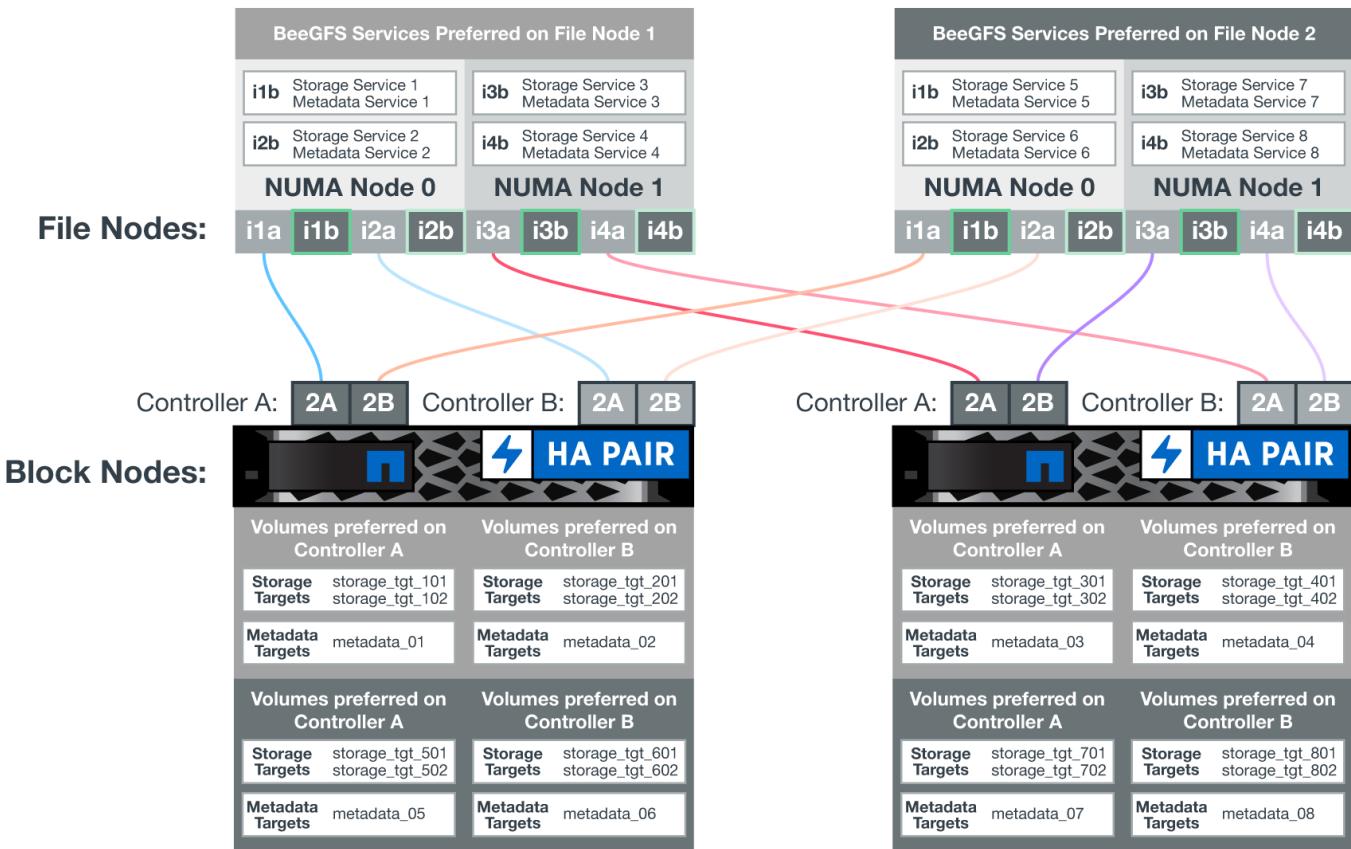
Los nodos de bloques constan de dos controladoras RAID activo/activo con acceso compartido al mismo conjunto de unidades. Por lo general, cada controladora tiene la mitad de los volúmenes configurados en el sistema, pero puede sustituir la otra controladora según sea necesario.

El software multivía en los nodos de archivos determina la ruta activa y optimizada para cada volumen y se mueve automáticamente a la ruta alternativa en caso de que se produzca un fallo en un cable, un adaptador o una controladora.

El siguiente diagrama muestra el diseño de la controladora en los nodos de bloques de EF600.



Para facilitar la solución de alta disponibilidad de disco compartido, los volúmenes se asignan a los dos nodos de archivo para que puedan hacerse cargo entre sí según sea necesario. En el siguiente diagrama se muestra un ejemplo de cómo se configura el servicio BeeGFS y la propiedad de volumen preferida para obtener el máximo rendimiento. La interfaz a la izquierda de cada servicio BeeGFS indica la interfaz preferida que los clientes y otros servicios utilizan para ponerse en contacto con él.



En el ejemplo anterior, los clientes y los servicios de servidor prefieren comunicarse con el servicio de almacenamiento 1 mediante la interfaz i1b. El servicio de almacenamiento 1 utiliza la interfaz i1a como ruta preferida para comunicarse con sus volúmenes (Storage_tgt_101, 102) en la controladora A del primer nodo de bloque. Esta configuración hace uso del ancho de banda PCIe bidireccional completo disponible para el adaptador InfiniBand y logra un mejor rendimiento a partir de un adaptador HDR InfiniBand de doble puerto del que sería posible con PCIe 4.0 de otro modo.

Configuración de nodos de archivos BeeGFS

Los nodos de archivos BeeGFS se configuran en un clúster de alta disponibilidad para facilitar la comutación por error de los servicios BeeGFS entre varios nodos de archivos.

El diseño de clúster de alta disponibilidad se basa en dos proyectos de alta disponibilidad de Linux ampliamente utilizados: Corosync para la pertenencia a clústeres y Pacemaker para la administración de recursos de clúster. Para obtener más información, consulte ["Formación de Red Hat para complementos de alta disponibilidad"](#).

NetApp es autor y creó varios agentes de recursos de marco de clúster abierto (OCF) ampliados para permitir que el clúster inicie y supervise de forma inteligente los recursos de BeeGFS.

Clústeres de alta disponibilidad de BeeGFS

Normalmente, cuando se inicia un servicio BeeGFS (con o sin ha), deben existir algunos recursos:

- Direcciones IP donde se puede acceder al servicio, generalmente configuradas por Network Manager.
- Sistemas de archivos subyacentes utilizados como objetivos para BeeGFS para almacenar datos.

Normalmente se definen en /etc/fstab Y montado por systemd.

- Servicio de sistema responsable de iniciar los procesos de BeeGFS cuando los otros recursos están listos.

Sin software adicional, estos recursos solo comienzan en un único nodo de archivo. Por lo tanto, si el nodo de archivo se desconecta, una parte del sistema de archivos BeeGFS no está accesible.

Debido a que varios nodos pueden iniciar cada servicio BeeGFS, Pacemaker debe asegurarse de que cada servicio y los recursos dependientes sólo se ejecutan en un nodo cada vez. Por ejemplo, si dos nodos intentan iniciar el mismo servicio BeeGFS, existe el riesgo de que se dañen los datos si ambos intentan escribir en los mismos archivos en el destino subyacente. Para evitar esta situación, Pacemaker confía en Corosync para mantener de forma fiable el estado general del clúster sincronizado entre todos los nodos y establecer quórum.

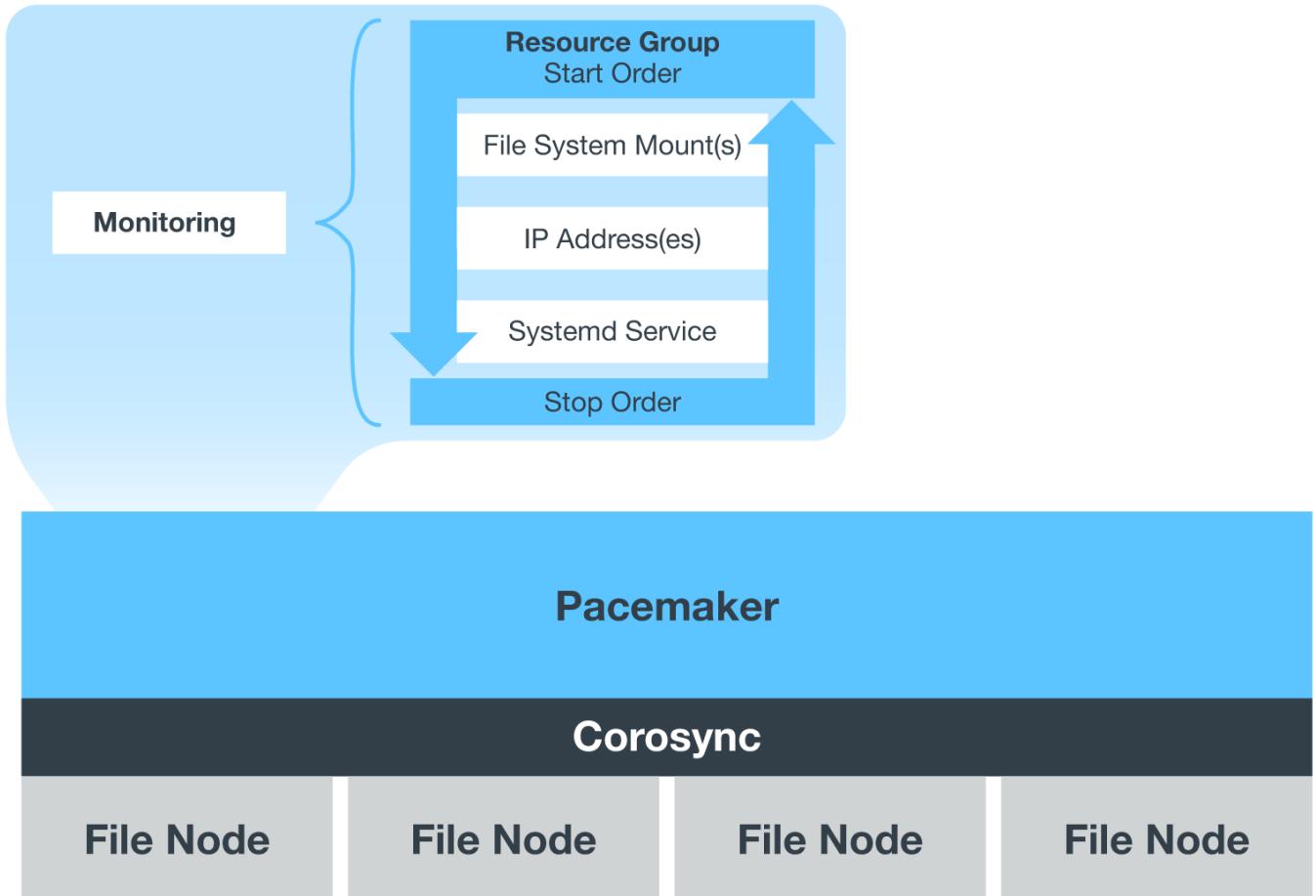
Si se produce un fallo en el clúster, Pacemaker reacciona y reinicia los recursos de BeeGFS en otro nodo. En algunos casos, es posible que Pacemaker no pueda comunicarse con el nodo defectuoso original para confirmar que los recursos están detenidos. Para verificar que el nodo está inactivo antes de reiniciar los recursos de BeeGFS en otra parte, Pacemaker apaga el nodo defectuoso, lo que es ideal para eliminar la alimentación.

Hay muchos agentes de esgrima de código abierto disponibles que permiten a Pacemaker cercar un nodo con una unidad de distribución de energía (PDU) o utilizando el controlador de administración de placa base del servidor (BMC) con API como Redfish.

Cuando BeeGFS se ejecuta en un clúster ha, Pacemaker gestiona todos los servicios BeeGFS y los recursos subyacentes en grupos de recursos. Cada servicio BeeGFS y los recursos de los que depende, se configuran en un grupo de recursos, que garantiza que los recursos se inician y se detienen en el orden correcto y se encuentran en el mismo nodo.

Para cada grupo de recursos BeeGFS, Pacemaker ejecuta un recurso de supervisión BeeGFS personalizado que es responsable de detectar condiciones de fallo y de activar de forma inteligente recuperaciones tras fallos cuando un servicio BeeGFS ya no está accesible en un nodo concreto.

La siguiente figura muestra los servicios y dependencias de BeeGFS controlados por marcapasos.



De modo que se inicien varios servicios BeeGFS del mismo tipo en el mismo nodo, Pacemaker se configura para iniciar servicios BeeGFS mediante el método de configuración Multi Mode. Para obtener más información, consulte ["Documentación de BeeGFS sobre modo múltiple"](#).

Debido a que los servicios BeeGFS deben poder iniciarse en varios nodos, el archivo de configuración de cada servicio (normalmente ubicado en `/etc/beegfs`) se almacena en uno de los volúmenes E-Series utilizados como objetivo BeeGFS para ese servicio. Esto hace que la configuración junto con los datos de un servicio BeeGFS en particular sea accesible para todos los nodos que puedan necesitar ejecutar el servicio.

```
# 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
```

Verificación del diseño

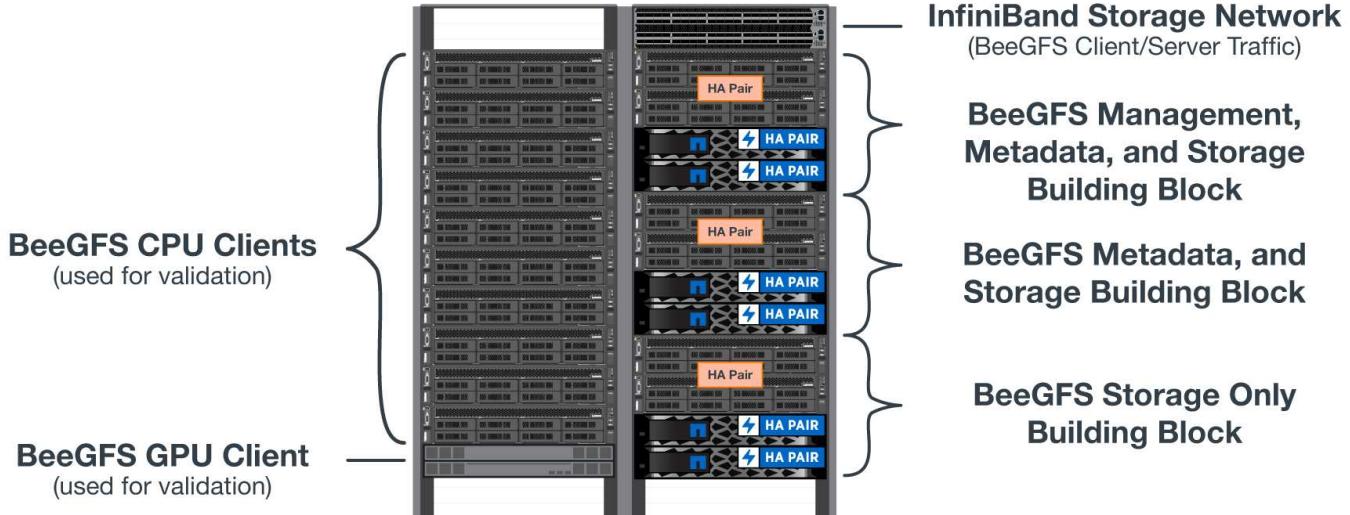
El diseño de segunda generación de BeeGFS en la solución de NetApp se verificó mediante tres perfiles de configuración de bloques básicos.

Los perfiles de configuración incluyen lo siguiente:

- Un único elemento básico, incluidos los servicios de gestión, metadatos y almacenamiento de BeeGFS.
- Metadatos BeeGFS más un elemento básico de almacenamiento.
- Un elemento básico de sólo almacenamiento BeeGFS.

Los elementos básicos se conectaron a dos switches NVIDIA Quantum InfiniBand (MQM8700). También se adjuntaron diez clientes BeeGFS a los switches InfiniBand y se utilizaron para ejecutar utilidades de análisis de rendimiento sintéticos.

En la siguiente figura, se muestra la configuración de BeeGFS que se utiliza para validar BeeGFS en la solución de NetApp.



Segmentación de archivos BeeGFS

Una ventaja de los sistemas de archivos paralelos es la capacidad de resegmentar archivos individuales en múltiples destinos de almacenamiento, lo que podría representar volúmenes en los mismos sistemas de almacenamiento subyacentes o en diferentes.

En BeeGFS, puede configurar la segmentación por directorio y por archivo para controlar el número de destinos utilizados para cada archivo y para controlar el tamaño de bloque (o el tamaño de bloque) utilizado para cada franja de archivo. Esta configuración permite al sistema de archivos admitir distintos tipos de cargas de trabajo y perfiles de I/O sin necesidad de reconfigurar o reiniciar los servicios. Puede aplicar la configuración de franja mediante `beegfs-ctl` Herramienta de línea de comandos o con aplicaciones que usan la API de segmentación. Para obtener más información, consulte la documentación de BeeGFS para "["Segmentación"](#)" y.. "["API de segmentación"](#)".

Para lograr el mejor rendimiento, los patrones de franjas se ajustaron durante la prueba, y se señalan los parámetros utilizados para cada prueba.

Pruebas de ancho de banda IOR: Múltiples clientes

Las pruebas de ancho de banda IOR utilizaban OpenMPI para ejecutar trabajos paralelos de la herramienta de generador de E/S sintético IOR (disponible en ["GitHub de HPC"](#)) A través de los 10 nodos de cliente a uno o más bloques de creación de BeeGFS. A menos que se indique lo contrario:

- Todas las pruebas utilizaron E/S directa con un tamaño de transferencia 1MiB.
- La segmentación de archivos BeeGFS se ha establecido en un tamaño de archivo de 1 MB y un objetivo por archivo.

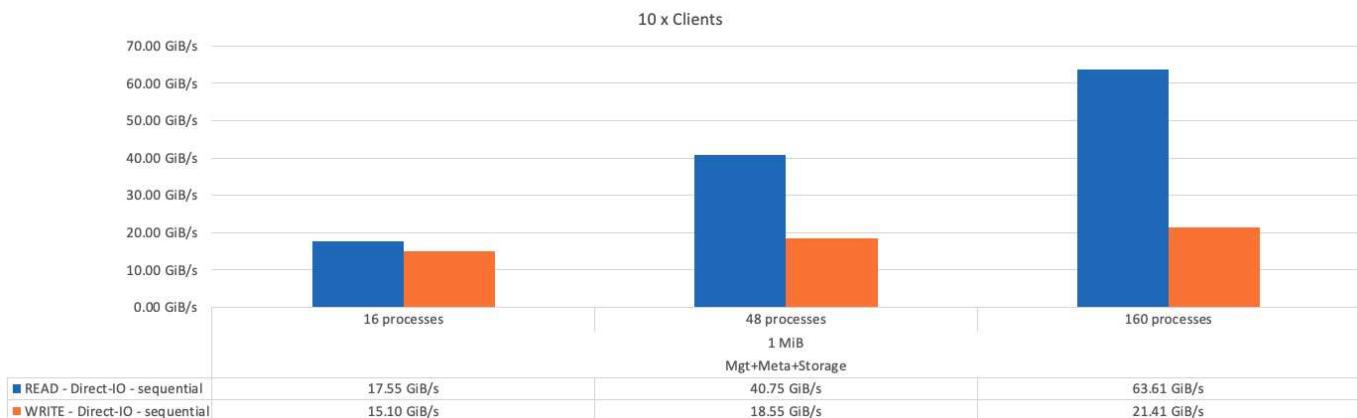
Se utilizaron los siguientes parámetros para IOR con el recuento de segmentos ajustado para mantener el tamaño del archivo agregado a 5 TiB para un bloque básico y 40 TiB para tres bloques básicos.

```
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
```

Un elemento básico de BeeGFS (gestión, metadatos y almacenamiento)

En la siguiente figura, se muestran los resultados de la prueba IOR con un solo elemento básico de BeeGFS

(gestión, metadatos y almacenamiento).



Metadatos BeeGFS + elemento básico de almacenamiento

En la siguiente figura se muestran los resultados de la prueba IOR con un único bloque de creación de almacenamiento y metadatos BeeGFS.



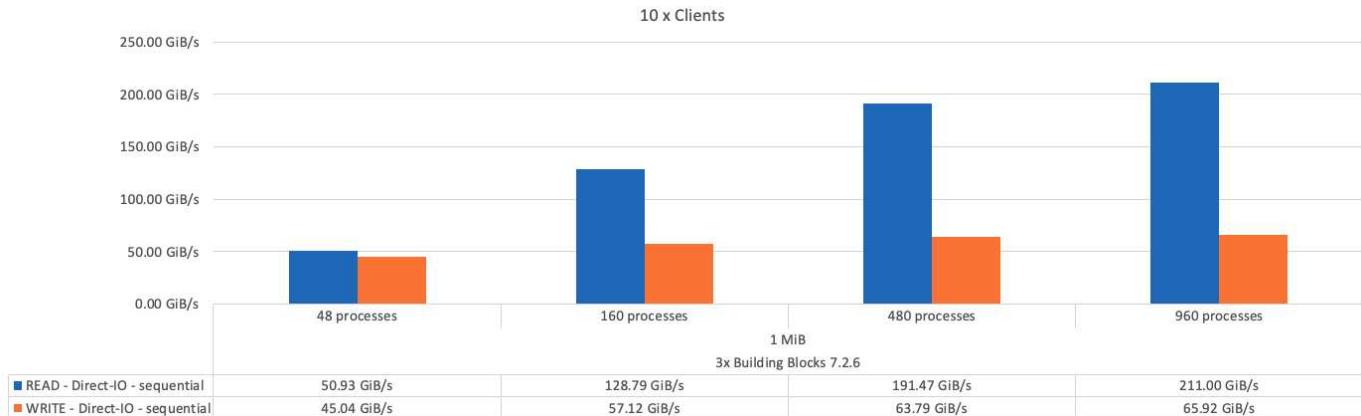
Elemento básico de sólo almacenamiento BeeGFS

En la siguiente figura se muestran los resultados de la prueba IOR con un solo elemento básico de almacenamiento BeeGFS.



Tres elementos básicos de BeeGFS

En la siguiente figura se muestran los resultados de la prueba IOR con tres bloques de construcción BeeGFS.



Según lo esperado, la diferencia de rendimiento entre el bloque básico y el bloque básico de metadatos + almacenamiento posterior es mínima. Si comparamos el elemento básico de metadatos + almacenamiento con un elemento básico exclusivo del almacenamiento, el rendimiento de lectura se aprecia un ligero aumento en el rendimiento de lectura debido a las unidades adicionales utilizadas como destino del almacenamiento. Sin embargo, no existe una diferencia significativa en el rendimiento de escritura. Para lograr un mayor rendimiento, puede añadir varios elementos básicos juntos para escalar el rendimiento de forma lineal.

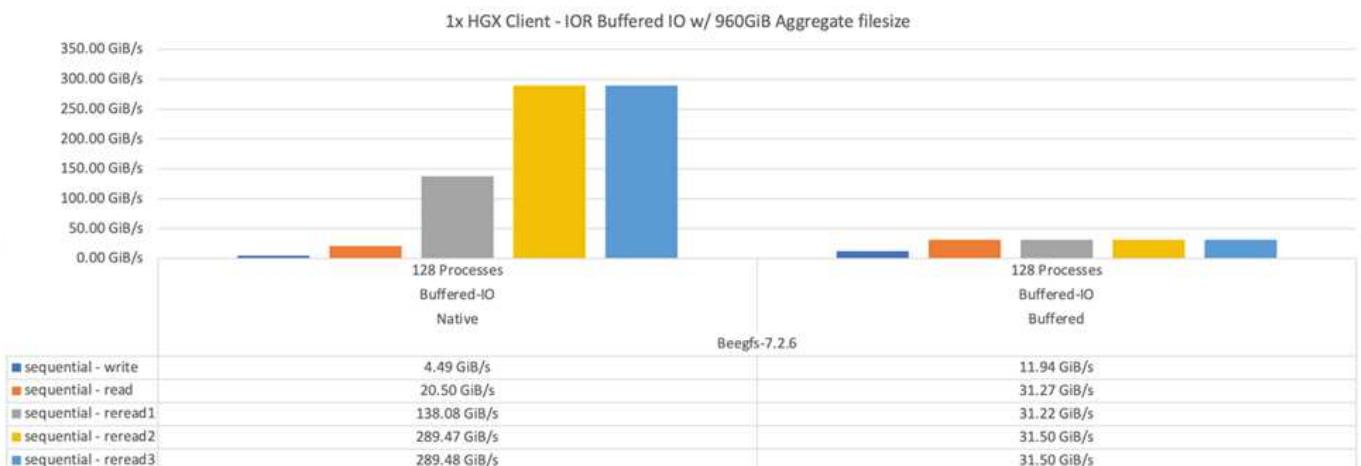
Pruebas de ancho de banda IOR: Un único cliente

La prueba de ancho de banda IOR utilizó OpenMPI para ejecutar varios procesos IOR utilizando un único servidor GPU de alto rendimiento para explorar el rendimiento que se puede obtener en un único cliente.

En esta prueba también se compara el comportamiento y el rendimiento de BeeGFS cuando el cliente está configurado para utilizar la caché de páginas del kernel de Linux (`tuneFileCacheType = native`) frente al valor predeterminado `buffered` ajuste.

El modo de almacenamiento en caché nativo utiliza la memoria caché de página del kernel de Linux en el cliente, lo que permite que las operaciones de nueva lectura provengan de la memoria local en lugar de retransmitirse a través de la red.

En el siguiente diagrama se muestran los resultados de las pruebas IOR con tres bloques de creación BeeGFS y un único cliente.



La segmentación de BeeGFS para estas pruebas se estableció en un tamaño de archivo de 1 MB con ocho objetivos por archivo.

Aunque el rendimiento de escritura y lectura inicial es superior mediante el modo de búfer predeterminado, en el caso de cargas de trabajo que releer los mismos datos varias veces, se produce un aumento significativo del rendimiento en el modo de almacenamiento en caché nativo. Este rendimiento mejorado de nueva obtención es importante para cargas de trabajo como el aprendizaje profundo que rellenan el mismo conjunto de datos varias veces a lo largo de muchas épocas.

Prueba de rendimiento de metadatos

En las pruebas de rendimiento de metadatos se utilizó la herramienta MDTest (incluida como parte de IOR) para medir el rendimiento de los metadatos de BeeGFS. Las pruebas utilizaron OpenMPI para ejecutar trabajos paralelos en los diez nodos cliente.

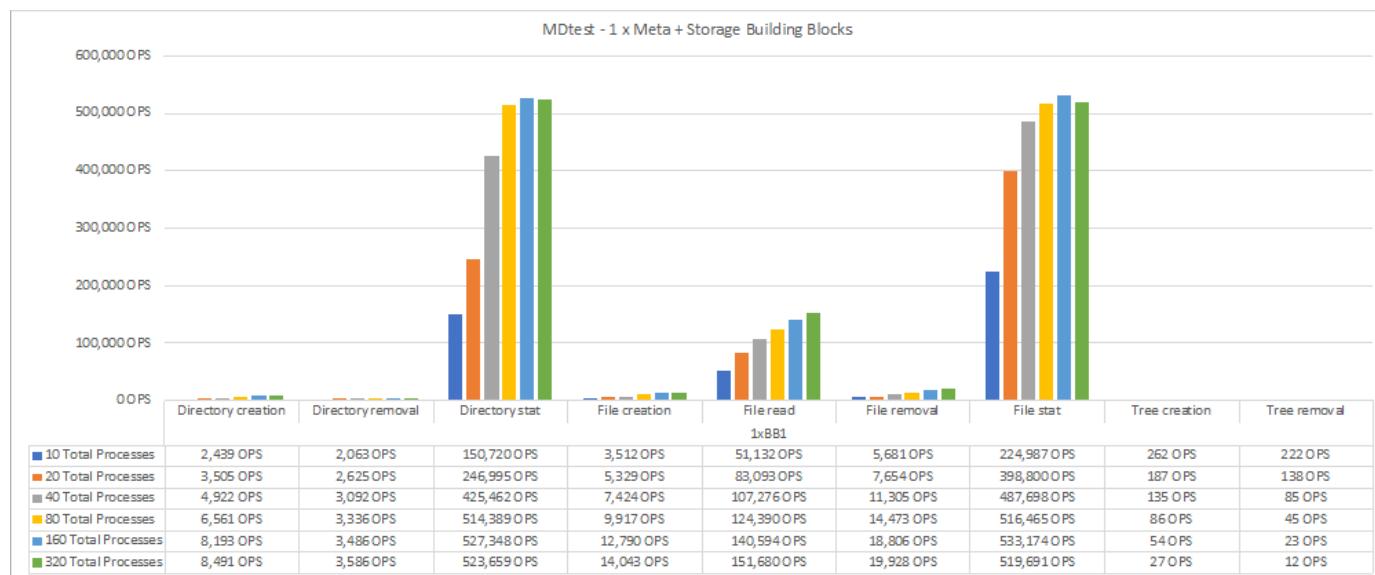
Se utilizaron los siguientes parámetros para ejecutar la prueba de referencia con el número total de procesos escalados de 10 a 320 en el paso del doble y con un tamaño de archivo de 4k.

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

El rendimiento de los metadatos se midió primero con uno entonces dos metadatos + elementos básicos del almacenamiento, para mostrar cómo se escala el rendimiento añadiendo elementos básicos adicionales.

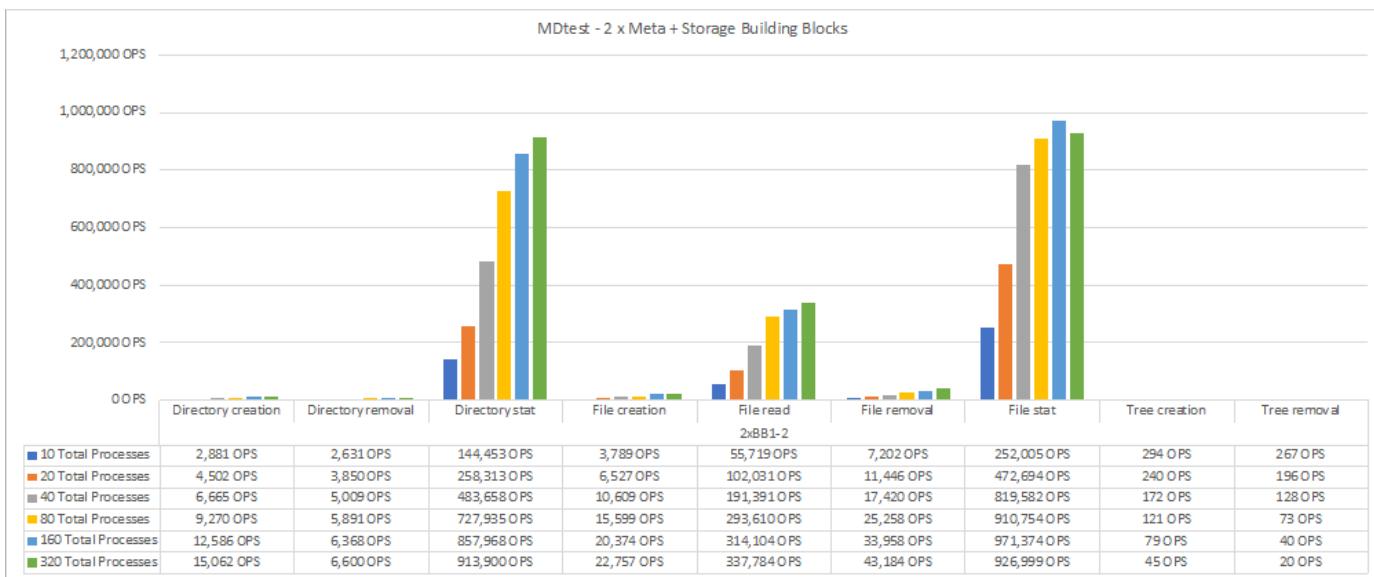
Un elemento básico de metadatos BeeGFS + almacenamiento

En el siguiente diagrama se muestran los resultados de MDTest con un bloque de creación de almacenamiento y metadatos BeeGFS.



Dos metadatos BeeGFS + elementos básicos de almacenamiento

El siguiente diagrama muestra los resultados de MDTest con dos metadatos BeeGFS + bloques de almacenamiento.



Validación funcional

Como parte de la validación de esta arquitectura, NetApp ejecutó varias pruebas funcionales incluyendo las siguientes:

- Al producirse un fallo en un puerto InfiniBand de un único cliente, se deshabilita el puerto del switch.
- Al producirse un fallo en un puerto InfiniBand de un único servidor, se deshabilita el puerto del switch.
- Activación de un apagado inmediato del servidor mediante el BMC.
- Colocación dignidad de un nodo en espera y commutación por error al servicio en otro nodo.
- Con dignidad, volver a colocar un nodo en línea y devolver servicios al nodo original.
- Apague uno de los switches InfiniBand mediante la PDU. Todas las pruebas se realizaron mientras las pruebas de estrés estaban en curso con el sysSessionChecksEnabled: false Parámetro definido en los clientes BeeGFS. No se han observado errores ni interrupciones en I/O.

i Hay un problema conocido (consulte "[Cambios](#)") Cuando las conexiones RDMA cliente/servidor BeeGFS se interrumpen inesperadamente, ya sea a través de la pérdida de la interfaz primaria (como se define en connInterfacesFile) O un servidor BeeGFS falla; la E/S de cliente activa se puede bloquear durante un máximo de diez minutos antes de continuar. Este problema no ocurre cuando los nodos BeeGFS se colocan correctamente dentro y fuera del modo de espera para el mantenimiento planificado o si TCP está en uso.

Validación de NVIDIA DGX SuperPOD y BasePOD

NetApp validó una solución de almacenamiento para nVIDIAS DGX A100 SuperPOD que utiliza un sistema de archivos BeeGFS similar que consiste en tres elementos básicos con los metadatos más el perfil de configuración de almacenamiento aplicado. El esfuerzo de cualificación incluyó probar la solución descrita en este NVA con veinte servidores DGX A100 GPU que ejecutan una gran variedad de pruebas de rendimiento de almacenamiento, aprendizaje automático y aprendizaje profundo. Basándose en la validación establecida con DGX A100 SuperPOD de NVIDIA, la solución BeeGFS en NetApp ha sido aprobada para los sistemas DGX SuperPOD H100, H200 y B200. Esta extensión se basa en cumplir con las pruebas de rendimiento y los requisitos del sistema previamente establecidos según la validación con el servidor DGX A100 de NVIDIA.

Para obtener más información, consulte "[NVIDIA DGX SuperPOD con NetApp](#)" y.. "[DGX BasePOD de NVIDIA](#)".

Directrices de tamaño

La solución BeeGFS incluye recomendaciones sobre el rendimiento y el ajuste de la capacidad basadas en pruebas de verificación.

El objetivo de una arquitectura de elementos básicos es crear una solución de tamaño sencillo mediante la adición de varios elementos básicos para satisfacer los requisitos de un sistema BeeGFS concreto. Con las siguientes pautas, puede estimar la cantidad y los tipos de bloques de construcción de BeeGFS que se necesitan para cumplir los requisitos de su entorno.

Tenga en cuenta que estas estimaciones representan el mejor caso de rendimiento. Las aplicaciones de pruebas de rendimiento sintéticas se escriben y se utilizan para optimizar el uso de sistemas de archivos subyacentes de formas que las aplicaciones del mundo real podrían no.

Ajuste de tamaño del rendimiento

La siguiente tabla proporciona un ajuste del tamaño del rendimiento recomendado.

Perfil de configuración	1MiB lee	1MiB escribe
Metadatos + almacenamiento	62GiBps	21 GiBps
Solo almacenamiento	64 GiBps	21 GiBps

Las estimaciones de tamaño de la capacidad de metadatos se basan en la "regla general" según la cual 500 GB de capacidad son suficientes para aproximadamente 150 millones de archivos en BeeGFS. (Para obtener más información, consulte la documentación de BeeGFS para "["Requisitos del sistema"](#).)

El uso de funciones como las listas de control de acceso y el número de directorios y archivos por directorio también afecta a la rapidez con la que se consume el espacio de metadatos. Las estimaciones de la capacidad de almacenamiento dan cuenta de la capacidad de unidad utilizable junto con la sobrecarga de RAID 6 y XFS.

Configuración de la capacidad para metadatos + elementos básicos de almacenamiento

La siguiente tabla proporciona un tamaño de capacidad recomendado para metadatos, además de los elementos básicos de almacenamiento.

Tamaño de la unidad (2+2 RAID 1) grupos de volúmenes de metadatos	Capacidad de metadatos (cantidad de archivos)	Grupos de volúmenes de almacenamiento de tamaño de unidad (8+2 RAID 6)	Capacidad de almacenamiento (contenido de archivos)
1,92 TB	1,938,577,200	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	17,269,854,000	15,3 TB	460 TB



Al ajustar el tamaño de los metadatos más los elementos básicos de almacenamiento, puede reducir los costes usando unidades más pequeñas para los grupos de volúmenes de metadatos frente a los grupos de volúmenes de almacenamiento.

Configuración de la capacidad para los elementos básicos solo del almacenamiento

La siguiente tabla proporciona ajuste de tamaño de capacidad de regla general para elementos básicos de solo almacenamiento.

Grupos de volúmenes de almacenamiento de tamaño de unidad (10+2 RAID 6)	Capacidad de almacenamiento (contenido de archivos)
1,92 TB	59,89 TB
3,84 TB	1319,80 TB
7,68 TB	251.89TB
15,3 TB	538,55 TB



La sobrecarga de rendimiento y capacidad de incluir el servicio de gestión en el elemento básico (primero) es mínima, a menos que se habilite el bloqueo global de archivos.

Ajuste del rendimiento

La solución BeeGFS incluye recomendaciones para el ajuste del rendimiento basadas en pruebas de verificación.

Si bien BeeGFS proporciona un rendimiento razonable desde el momento de su instalación, NetApp ha desarrollado un conjunto de parámetros de ajuste recomendados para maximizar el rendimiento. Estos parámetros tienen en cuenta las funcionalidades de los nodos de bloque E-Series subyacentes y todos los requisitos especiales necesarios para ejecutar BeeGFS en una arquitectura de alta disponibilidad de disco compartido.

Ajuste del rendimiento de los nodos de archivos

Los parámetros de ajuste disponibles que puede configurar son los siguientes:

- Configuración del sistema en el UEFI/BIOS de nodos de archivos.** para maximizar el rendimiento, recomendamos configurar los ajustes del sistema en el modelo de servidor que utilice como nodos de archivos. Los ajustes del sistema se configuran cuando se configuran los nodos de archivos mediante la configuración del sistema (UEFI/BIOS) o las API Redfish proporcionadas por el controlador de administración de la placa base (BMC).

La configuración del sistema varía en función del modelo de servidor que utilice como nodo de archivos. Los ajustes deben configurarse manualmente en función del modelo de servidor que se esté utilizando. Para aprender a configurar los ajustes del sistema para los nodos de archivos Lenovo SR665 V3 validados, consulte "["Ajuste la configuración del sistema del nodo de archivos para aumentar el rendimiento"](#)".

- Configuración predeterminada de los parámetros de configuración necesarios.** los parámetros de configuración necesarios afectan a la forma en que se configuran los servicios BeeGFS y cómo los volúmenes E-Series (dispositivos de bloques) se formatean y montan mediante Pacemaker. Entre estos parámetros de configuración necesarios se incluyen los siguientes:

- Parámetros de configuración del servicio BeeGFS

Es posible anular la configuración predeterminada para los parámetros de configuración según sea necesario. Para obtener los parámetros que puede ajustar a sus cargas de trabajo específicas o casos

de uso, consulte la "["Parámetros de configuración del servicio BeeGFS"](#)".

- El formato de los volúmenes y los parámetros de montaje se establecen en los valores predeterminados recomendados y solo se deben ajustar en casos prácticos avanzados. Los valores predeterminados harán lo siguiente:

- Optimización del formato de volumen inicial basado en el tipo de destino (como la gestión, los metadatos o el almacenamiento), junto con la configuración de RAID y el tamaño de segmentos del volumen subyacente.
- Ajuste cómo monta Pacemaker cada volumen para garantizar que los cambios se vacíen inmediatamente a los nodos de bloque E-Series. De este modo se evita la pérdida de datos cuando fallan nodos de archivos con las escrituras activas en curso.

Para obtener los parámetros que puede ajustar a sus cargas de trabajo específicas o casos de uso, consulte la "["parámetros de configuración de formato de volumen y montaje"](#)".

3. Configuración del sistema en el sistema operativo Linux instalado en los nodos de archivos.

Puede anular la configuración predeterminada del sistema operativo Linux al crear el inventario de Ansible en el paso 4 de "["Cree el inventario de Ansible"](#)".

La configuración predeterminada se utilizó para validar BeeGFS en la solución de NetApp, pero es posible modificarla para adaptarla a sus cargas de trabajo o casos de uso específicos. Algunos ejemplos de la configuración del sistema operativo Linux que puede cambiar son los siguientes:

- Las colas de I/o en dispositivos de bloques E-Series.

Se pueden configurar colas de I/o en los dispositivos de bloque E-Series que se utilizan como destinos BeeGFS para:

- Ajuste el algoritmo de programación en función del tipo de dispositivo (NVMe, HDD, etc.).
 - Aumentar el número de solicitudes pendientes.
 - Ajustar los tamaños de las solicitudes.
 - Optimice el comportamiento de lectura anticipada.
- Ajustes de memoria virtual.

Puede ajustar la configuración de memoria virtual para obtener un rendimiento de transmisión sostenido óptimo.

- Configuración de CPU.

Puede ajustar el regulador de frecuencia de la CPU y otras configuraciones de la CPU para obtener el máximo rendimiento.

- Tamaño de solicitud de lectura.

Puede aumentar el tamaño máximo de solicitud de lectura para las HCA de NVIDIA.

Ajuste del rendimiento para nodos de bloques

En función de los perfiles de configuración aplicados a un bloque de creación de BeeGFS en particular, los grupos de volúmenes configurados en los nodos de bloque cambian ligeramente. Por ejemplo, con un nodo de bloque EF600 de 24 unidades:

- Para el único elemento básico, incluidos los servicios de gestión, metadatos y almacenamiento de BeeGFS:
 - 1 grupo de volúmenes de 2+2 RAID 10 para servicios de metadatos y gestión de BeeGFS
 - 2 grupos de volúmenes RAID 6 de 8+2 para servicios de almacenamiento BeeGFS
- Para un bloque básico de metadatos BeeGFS + almacenamiento:
 - 1 grupo de volúmenes de 2+2 RAID 10 para servicios de metadatos BeeGFS
 - 2 grupos de volúmenes RAID 6 de 8+2 para servicios de almacenamiento BeeGFS
- Para el almacenamiento BeeGFS, solo elemento básico:
 - 2 grupos de volúmenes RAID 6 de 10+2 para servicios de almacenamiento BeeGFS

 Como BeeGFS necesita menos espacio de almacenamiento para la gestión y los metadatos en comparación con el almacenamiento, una opción es utilizar unidades más pequeñas para los grupos de volúmenes RAID 10. Las unidades más pequeñas deben llenarse en las ranuras de unidad más externas. Para obtener más información, consulte "["Instrucciones de puesta en funcionamiento"](#)".

Todos estos ajustes se configuran mediante la puesta en marcha basada en Ansible, junto con otros ajustes que suelen recomendarse para optimizar el rendimiento o el comportamiento, entre los que se incluyen:

- Ajustar el tamaño de bloque de caché global a 32 KiB y ajustar el vaciado de caché basado en demanda al 80 %.
- Al deshabilitar el equilibrio de carga automático (se garantiza que las asignaciones de volúmenes de la controladora permanezcan según la definición).
- Habilitar el almacenamiento en caché de lectura y deshabilitar el almacenamiento en caché de lectura anticipada.
- Habilitar el almacenamiento en caché de escritura con mirroring y requerir backup de batería, de modo que la caché se mantiene mediante el fallo de una controladora del nodo de bloque.
- Especificar el orden en que las unidades se asignan a grupos de volúmenes, equilibrando las operaciones de I/o en los canales de unidades disponibles.

Elemento básico de gran capacidad

El diseño de la solución BeeGFS estándar se ha diseñado teniendo en cuenta las cargas de trabajo de alto rendimiento. Los clientes que busquen casos de uso de gran capacidad deben observar las variaciones en las características de diseño y rendimiento descritas aquí.

Configuración de hardware y software

La configuración de hardware y software del elemento básico de alta capacidad es de serie, a excepción de que las controladoras EF600 deben sustituirse por una controladora EF300 con opción de conectarse entre 1 y 7 bandejas de expansión IOM con 60 unidades cada una para cada cabina de almacenamiento, un total de 2 a 14 bandejas de expansión por bloque de construcción.

Es probable que los clientes que implementan un diseño de elemento básico de gran capacidad utilicen solo la configuración del estilo de bloque de creación base compuesta por servicios de gestión, metadatos y almacenamiento de BeeGFS para cada nodo. Para reducir la rentabilidad, los nodos de almacenamiento de gran capacidad deben aprovisionar volúmenes de metadatos en las unidades NVMe en el compartimento de

controladora EF300 y deben aprovisionar volúmenes de almacenamiento a las unidades NL-SAS de las bandejas de expansión.



Directrices de tamaño

Estas directrices de tamaño suponen que los bloques básicos de gran capacidad se configuran con un grupo de volúmenes SSD de 2+2 NVMe para los metadatos en el compartimento EF300 básico y 6 grupos de volúmenes NL-SAS de 8+2 por bandeja de ampliación IOM para el almacenamiento.

Tamaño de unidad (HDD de capacidad)	Capacidad por BB (1 bandeja)	Capacidad por BB (2 bandejas)	Capacidad por BB (3 bandejas)	Capacidad por BB (4 bandejas)
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

Ponga en marcha la solución

Información general sobre la implementación

BeeGFS en NetApp puede ponerse en marcha en nodos de archivos y bloques validados utilizando Ansible con el diseño de elementos básicos BeeGFS de NetApp.

Colecciones y roles de Ansible

La solución BeeGFS en NetApp se pone en marcha usando Ansible, un motor de automatización TECNOLÓGICA muy conocido que automatiza las puestas en marcha de aplicaciones. Ansible utiliza una serie de archivos conocidos colectivamente como el inventario, que modela el sistema de archivos BeeGFS que desea implementar.

Ansible permite a empresas como NetApp ampliar su funcionalidad incorporada mediante colecciones disponibles en Ansible Galaxy (consulte "[Colección BeeGFS de NetApp E-Series](#)"). Las recopilaciones incluyen módulos que realizan funciones o tareas específicas (como crear un volumen E-Series) y roles que pueden llamar a varios módulos y otros roles. Este método automatizado reduce el tiempo necesario para poner en marcha el sistema de archivos BeeGFS y el clúster de alta disponibilidad subyacente. Además, simplifica el mantenimiento y expansión del clúster y el sistema de archivos BeeGFS.

Para obtener detalles adicionales, consulte "[Obtenga más información sobre el inventario de Ansible](#)".



Dado que existen numerosos pasos que se deben seguir en la puesta en marcha de BeeGFS en la solución de NetApp, NetApp no admite la puesta en marcha manual de la solución.

Perfiles de configuración para los bloques de creación de BeeGFS

Los procedimientos de implementación cubren los siguientes perfiles de configuración:

- Un elemento básico que incluye servicios de gestión, metadatos y almacenamiento.
- Un segundo elemento básico que incluye metadatos y servicios de almacenamiento.
- Un tercer elemento básico que únicamente incluye servicios de almacenamiento.

Estos perfiles muestran toda la gama de perfiles de configuración recomendados para los elementos básicos de BeeGFS de NetApp. Para cada puesta en marcha, el número de elementos básicos de metadatos y almacenamiento o de servicios de almacenamiento, o los elementos básicos de uso exclusivo de servicios de almacenamiento, pueden variar en función de los requisitos de capacidad y rendimiento.

Información general sobre los pasos de implementación

La implementación implica las siguientes tareas de alto nivel:

Puesta en marcha de hardware

1. Monte físicamente cada bloque.
2. Hardware para montaje en rack y cableado. Para conocer los procedimientos detallados, consulte "["Ponga en marcha el hardware"](#)".

Puesta en marcha de software

1. "["Configure los nodos de archivos y bloques"](#)".
 - Configure las IP de BMC en los nodos de archivo
 - Instale un sistema operativo compatible y configure las redes de gestión en los nodos de archivos
 - Configure las IP de gestión en los nodos de bloques
2. "["Configure un nodo de control Ansible"](#)".
3. "["Ajuste la configuración del sistema para obtener rendimiento"](#)".
4. "["Cree el inventario de Ansible"](#)".
5. "["Defina el inventario de Ansible para los bloques de creación de BeeGFS"](#)".
6. "["Ponga en marcha BeeGFS con Ansible"](#)".
7. "["Configurar clientes BeeGFS"](#)".

Los procedimientos de implementación incluyen varios ejemplos en los que el texto debe copiarse a un archivo. Preste mucha atención a cualquier comentario en línea indicado por los caracteres "#" o "//" para cualquier cosa que deba o pueda modificarse para una implementación específica. Por ejemplo:



```
`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" `
```

Arquitecturas derivadas con variaciones en las recomendaciones de puesta en marcha:

- "["Elementos básicos de alta capacidad"](#)"

Obtenga más información sobre el inventario de Ansible

Antes de iniciar una implementación, familiarícese con cómo se configura y utiliza Ansible para poner en marcha BeeGFS en la solución NetApp.

El inventario de Ansible es una estructura de directorios que enumera los nodos de archivos y bloques del sistema de archivos BeeGFS en el que se va a poner en marcha. Incluye hosts, grupos y variables que describen el sistema de archivos BeeGFS deseado. El inventario de Ansible se tiene que almacenar en el nodo de control de Ansible, que es cualquier máquina con acceso al archivo y los nodos de bloque que se utilizan para ejecutar el libro de estrategia de Ansible. Los inventarios de muestra se pueden descargar de la ["E-Series BeeGFS GitHub de NetApp"](#).

Módulos y roles de Ansible

Para aplicar la configuración descrita en el inventario de Ansible, utilice los distintos módulos y roles de Ansible que se proporcionan en la colección Ansible de E-Series de NetApp (disponible en ["E-Series BeeGFS GitHub de NetApp"](#)), que ponen en marcha la solución completa.

Cada rol de la colección de Ansible de E-Series de NetApp es una puesta en marcha completa de BeeGFS en una solución de NetApp. Los roles utilizan las colecciones SANtricity, host y BeeGFS de E-Series de NetApp que permiten configurar el sistema de archivos BeeGFS con alta disponibilidad (alta disponibilidad). Luego, podrá aprovisionar y asignar almacenamiento, y garantizar que el almacenamiento del clúster esté listo para su uso.

Aunque se proporciona documentación en profundidad con los roles, los procedimientos de implementación describen cómo usar el rol para implementar una arquitectura verificada de NetApp mediante el diseño de elementos básicos BeeGFS de segunda generación.

 Aunque los pasos de puesta en marcha intentan proporcionar información suficiente para que la experiencia previa con Ansible no sea un requisito previo, debe tener algo de familiaridad con Ansible y la terminología relacionada.

Diseño de inventario para un clúster de alta disponibilidad de BeeGFS

Defina un clúster de alta disponibilidad de BeeGFS mediante la estructura de inventario de Ansible.

Cualquier persona con experiencia previa en Ansible debería tener en cuenta que el rol de alta disponibilidad BeeGFS implementa un método personalizado para detectar qué variables (o hechos) se aplican a cada host. Este diseño simplifica la estructuración del inventario de Ansible para describir los recursos que pueden ejecutarse en varios servidores.

Un inventario de Ansible normalmente se compone de los archivos de `host_vars` y `group_vars`, junto con un `inventory.yml` archivo que asigna hosts a grupos específicos (y potencialmente grupos a otros grupos).

 No cree ningún archivo con el contenido de esta subsección, que se piensa sólo como ejemplo.

A pesar de que esta configuración se basa por predeterminado en el perfil de configuración, debe tener un conocimiento general de cómo se presenta todo como un inventario de Ansible, tal y como se indica a continuación:

```

# 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.

```

Para cada servicio, se crea un archivo adicional en `group_vars` descripción de su configuración:

```

# 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

```

Este diseño permite definir el servicio, la red y la configuración de almacenamiento de BeeGFS para cada recurso en un único lugar. En segundo plano, el rol BeeGFS agrega la configuración necesaria para cada nodo de archivo y bloque basándose en esta estructura de inventario.

 El código numérico y el ID de nodo de cadena de BeeGFS para cada servicio se configuran automáticamente en función del nombre del grupo. Por lo tanto, además del requisito general de Ansible para que los nombres de grupo sean únicos, los grupos que representan un servicio BeeGFS deben finalizar en un número único para el tipo de servicio BeeGFS que representa el grupo. Por ejemplo, se permiten meta_01 y stor_01, pero los metadatos_01 y meta_01 no lo están.

Revise las prácticas recomendadas

Siga las directrices de prácticas recomendadas para poner en marcha BeeGFS en una solución de NetApp.

Convenciones estándar

Al ensamblar y crear físicamente el archivo de inventario de Ansible, siga estas convenciones estándar (para obtener más información, consulte "[Cree el inventario de Ansible](#)").

- Los nombres de host de los nodos de archivos se numeran secuencialmente (h01-HN) con números inferiores en la parte superior del rack y números superiores en la parte inferior.

Por ejemplo, la convención de nomenclatura [location] [row] [rack] hN se parece a beegfs_01:..

- Cada nodo de bloques se compone de dos controladoras de almacenamiento, cada una con su propio nombre de host.

Se utiliza el nombre de una cabina de almacenamiento para hacer referencia a todo el sistema de almacenamiento basado en bloques como parte de un inventario de Ansible. Los nombres de las cabinas de almacenamiento deben numerarse secuencialmente (a01 - an), y los nombres de host para las controladoras individuales provienen de esa convención de nomenclatura.

Por ejemplo, un nodo de bloque llamado `ictad22a01` normalmente puede tener nombres de host configurados para cada controladora como `ictad22a01-a` y `ictad22a01-b`, pero se puede consultar en un inventario de Ansible como `netapp_01`.

- Los nodos de archivo y de bloque dentro del mismo bloque básico comparten el mismo esquema de numeración y están adyacentes uno al otro en el rack con ambos nodos de archivo en la parte superior y ambos nodos de bloque directamente debajo de ellos.

Por ejemplo, en el primer bloque de creación, los nodos de archivo h01 y h02 están conectados directamente a los nodos de bloque a01 y a02. De arriba a abajo, los nombres de host son h01, h02, a01 y a02.

- Los bloques de creación se instalan en orden secuencial en función de sus nombres de host, de modo que los nombres de host con el número inferior se encuentran en la parte superior del rack y los nombres de host con un número superior se encuentran en la parte inferior.

El objetivo es minimizar la longitud del cable que se ejecuta en la parte superior de los switches del bastidor y definir una práctica de implementación estándar para simplificar la solución de problemas. En los centros de datos en los que no se permite esto debido a preocupaciones en torno a la estabilidad del rack, la inversa está permitida, rellenando el rack desde la parte inferior hacia arriba.

Configuración de la red de almacenamiento InfiniBand

La mitad de los puertos InfiniBand de cada nodo de archivos se utilizan para conectarse directamente a los nodos de bloques. La otra mitad está conectada a los switches InfiniBand y se utiliza para la conectividad cliente-servidor BeeGFS. Al determinar el tamaño de las subredes IPoIB que se utilizan para los clientes y servidores de BeeGFS, debe tener en cuenta el crecimiento previsto del clúster de computación/GPU y del sistema de archivos BeeGFS. Si debe desviarse de los rangos de IP recomendados, tenga en cuenta que cada conexión directa en un único bloque de creación tiene una subred única y que no se solapan con las subredes utilizadas para la conectividad cliente-servidor.

Conecciones directas

Los nodos de archivo y bloque dentro de cada elemento básico siempre utilizan las direcciones IP de la tabla siguiente para sus conexiones directas.



Este esquema de direccionamiento se adhiere a la siguiente regla: El tercer octeto siempre es impar o incluso, que depende de si el nodo de archivo es impar o par.

Nodo de archivo	Puerto IB	Dirección IP	Nodo de bloques	Puerto IB	IP física	IP virtual
Impar (h1)	i1a	192.168.1.10	Impar (c1)	2 a	192.168.1.100	192.168.1.101
Impar (h1)	i2a	192.168.3.10	Impar (c1)	2 a	192.168.3.100	192.168.3.101

Nodo de archivo	Puerto IB	Dirección IP	Nodo de bloques	Puerto IB	IP física	IP virtual
Impar (h1)	i3a	192.168.5.10	Par (c2)	2 a	192.168.5.100	192.168.5.101
Impar (h1)	i4a	192.168.7.10	Par (c2)	2 a	192.168.7.100	192.168.7.101
Par (h2)	i1a	192.168.2.10	Impar (c1)	2b	192.168.2.100	192.168.2.101
Par (h2)	i2a	192.168.4.10	Impar (c1)	2b	192.168.4.100	192.168.4.101
Par (h2)	i3a	192.168.6.10	Par (c2)	2b	192.168.6.100	192.168.6.101
Par (h2)	i4a	192.168.8.10	Par (c2)	2b	192.168.8.100	192.168.8.101

Esquemas de direccionamiento IPoIB cliente-servidor BeeGFS

Cada nodo de archivos ejecuta varios servicios de servidor BeeGFS (gestión, metadatos o almacenamiento). Para permitir que cada servicio commute al nodo de archivos de manera independiente, cada uno de ellos está configurado con direcciones IP únicas que pueden flotarse entre ambos nodos (a veces denominados una interfaz lógica o LIF).

Aunque no es obligatorio, esta implementación presupone que los siguientes rangos de subred IPoIB están en uso para estas conexiones y define un esquema de direccionamiento estándar que aplica las siguientes reglas:

- El segundo octeto siempre es impar o par, según si el puerto InfiniBand del nodo de archivo es impar o par.
- Las IP del clúster de BeeGFS siempre lo son xxx. 127.100.yyy o. xxx.128.100.yyy.

 Además de la interfaz utilizada para la gestión del SO en banda, Corosync puede utilizar interfaces adicionales para la sincronización y la golpiza de corazón en cluster. De este modo se garantiza que la pérdida de una única interfaz no apague todo el clúster.

- El servicio de gestión de BeeGFS siempre está en xxx.yyy.101.0 o. xxx.yyy.102.0.
- Los servicios de metadatos de BeeGFS siempre están en xxx.yyy.101.zzz o. xxx.yyy.102.zzz.
- Los servicios de almacenamiento de BeeGFS siempre están en xxx.yyy.103.zzz o xxx.yyy.104.zzz.
- Direcciones del intervalo 100.xxx.1.1 por 100.xxx.99.255 están reservados para clientes.

Esquema de direccionamiento de subred única IPoIB

Esta guía de despliegue utilizará un único esquema de subred dadas las ventajas enumeradas en "arquitectura de software".

Subred: 100.127.0.0/16

La siguiente tabla proporciona el rango para una sola subred: 100.127.0.0/16.

Específico	Puerto InfiniBand	Dirección IP o rango
IP de clúster de BeeGFS	i1b o i4b	100.127.100.1 - 100.127.100.255

Específico	Puerto InfiniBand	Dirección IP o rango
Gestión de BeeGFS	i1b	100.127.101.0
	i2b	100.127.102.0
Metadatos de BeeGFS	i1b o i3b	100.127.101.1 - 100.127.101.255
	i2b o i4b	100.127.102.1 - 100.127.102.255
Almacenamiento de BeeGFS	i1b o i3b	100.127.103.1 - 100.127.103.255
	i2b o i4b	100.127.104.1 - 100.127.104.255
Clientes BeeGFS	(varía según el cliente)	100.127.1.1 - 100.127.99.255

IPoIB Esquema de direccionamiento de dos subredes

Ya no se recomienda un esquema de direccionamiento de dos subredes, pero aún se puede implementar. Consulte las siguientes tablas para ver un esquema de dos subredes recomendado.

Subred A: 100.127.0.0/16

En la tabla siguiente se muestra el intervalo de la subred A: 100.127.0.0/16.

Específico	Puerto InfiniBand	Dirección IP o rango
IP de clúster de BeeGFS	i1b	100.127.100.1 - 100.127.100.255
Gestión de BeeGFS	i1b	100.127.101.0
Metadatos de BeeGFS	i1b o i3b	100.127.101.1 - 100.127.101.255
Almacenamiento de BeeGFS	i1b o i3b	100.127.103.1 - 100.127.103.255
Clientes BeeGFS	(varía según el cliente)	100.127.1.1 - 100.127.99.255

Subred B: 100.128.0.0/16

En la tabla siguiente se muestra el intervalo para la subred B: 100.128.0.0/16.

Específico	Puerto InfiniBand	Dirección IP o rango
IP de clúster de BeeGFS	i4b	100.128.100.1 - 100.128.100.255
Gestión de BeeGFS	i2b	100.128.102.0
Metadatos de BeeGFS	i2b o i4b	100.128.102.1 - 100.128.102.255
Almacenamiento de BeeGFS	i2b o i4b	100.128.104.1 - 100.128.104.255
Clientes BeeGFS	(varía según el cliente)	100.128.1.1 - 100.128.99.255

 No todas las IP de los rangos anteriores se utilizan en esta arquitectura verificada de NetApp. Muestran cómo se pueden preasignar direcciones IP para permitir una sencilla expansión del sistema de archivos mediante un esquema de direccionamiento IP coherente. En este esquema, los nodos de archivo BeeGFS y los ID de servicio corresponden con el cuarto octeto de un rango conocido de IP. El sistema de archivos podría escalarse más allá de los 255 nodos o servicios si fuera necesario.

Ponga en marcha el hardware

Cada elemento básico consta de dos nodos de archivos x86 validados conectados directamente a dos nodos de bloque mediante cables InfiniBand HDR (200 GB).



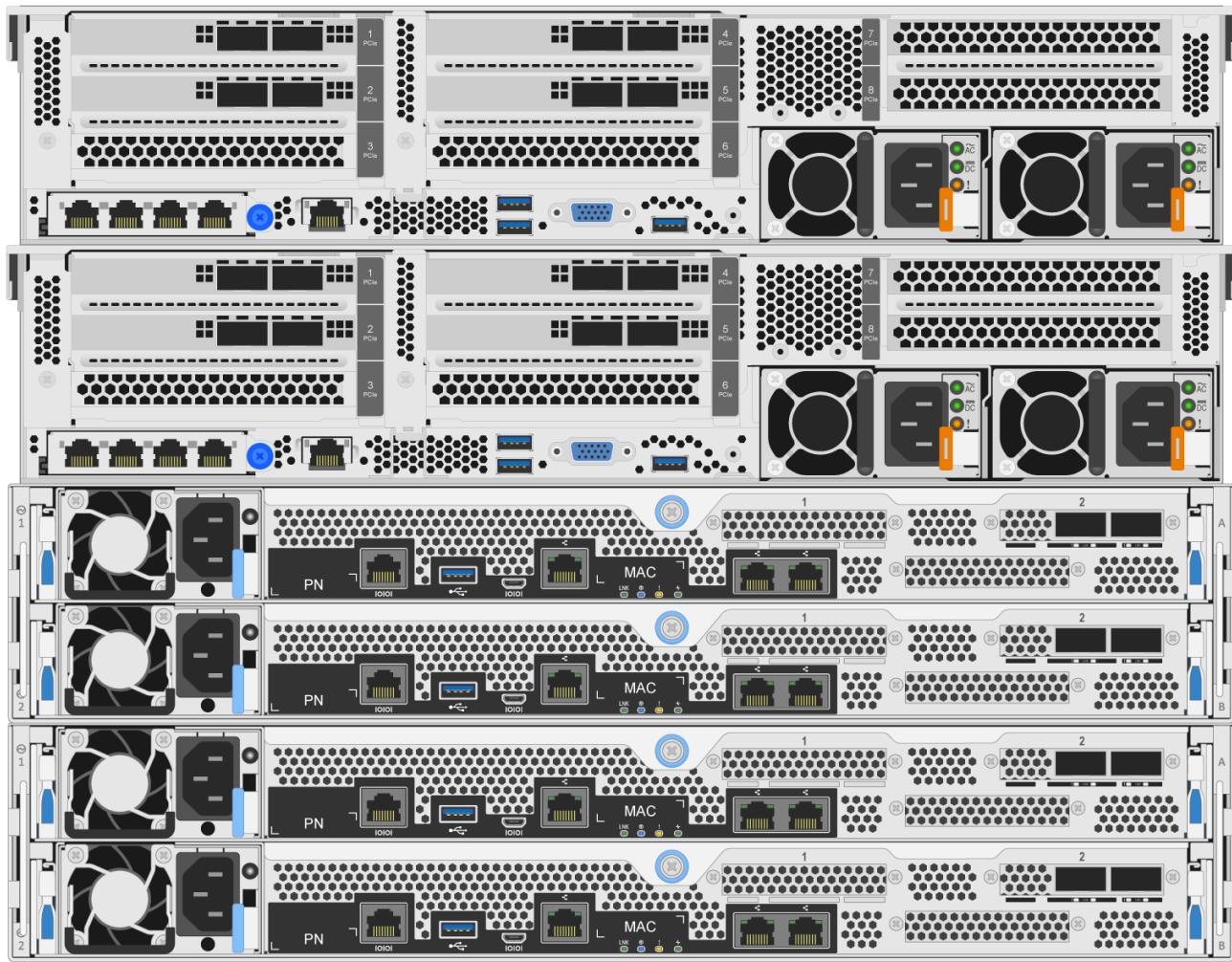
Se necesita un mínimo de dos bloques de construcción para establecer el quórum en el clúster de conmutación por error. Un clúster de dos nodos tiene limitaciones que podrían impedir que se produzca una conmutación al respaldo correcta. Puede configurar un clúster de dos nodos incorporando un tercer dispositivo como tiebreaker; sin embargo, esta documentación no describe ese diseño.

Los siguientes pasos son idénticos para cada elemento básico del clúster, independientemente de si se utiliza para ejecutar servicios de metadatos y almacenamiento de BeeGFS, o solo servicios de almacenamiento, a menos que se especifique lo contrario.

Pasos

1. Configure cada nodo de archivo BeeGFS con cuatro adaptadores de canal de host (HCA) utilizando los modelos especificados en la "[Requisitos técnicos](#)". Inserte las HCA en las ranuras PCIe del nodo de archivo de acuerdo con las especificaciones siguientes:
 - **Servidor Lenovo ThinkSystem SR665 V3:** Utilice las ranuras PCIe 1, 2, 4 y 5.
 - **Servidor Lenovo ThinkSystem SR665:** Utilice las ranuras PCIe 2, 3, 5 y 6.
2. Configure cada nodo de bloque BeeGFS con una tarjeta de interfaz del host (HIC) de 200 GB de puerto doble e instale la HIC en cada una de sus dos controladoras de almacenamiento.

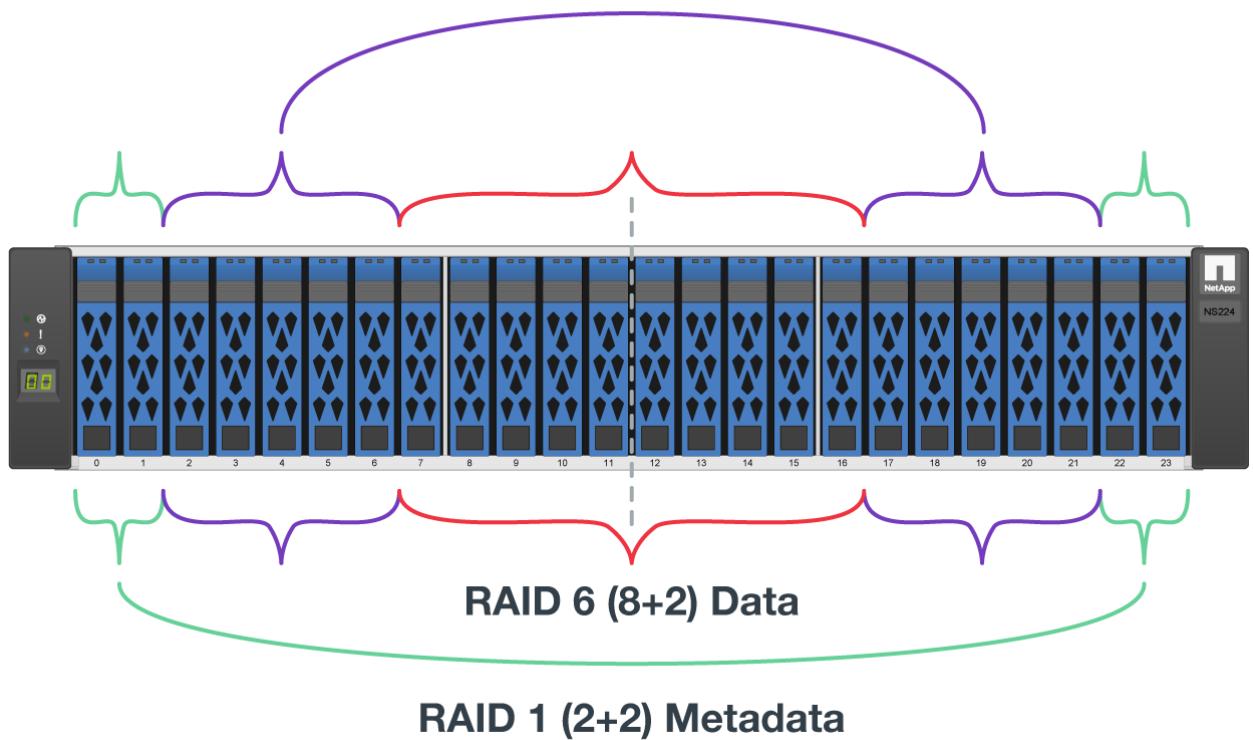
Monte en rack los bloques de creación de forma que los dos nodos de archivo BeeGFS se encuentren por encima de los nodos de bloque BeeGFS. La siguiente figura muestra la configuración correcta del hardware para el bloque de construcción BeeGFS que utiliza los servidores Lenovo ThinkSystem SR665 V3 como nodos de archivo (vista posterior).



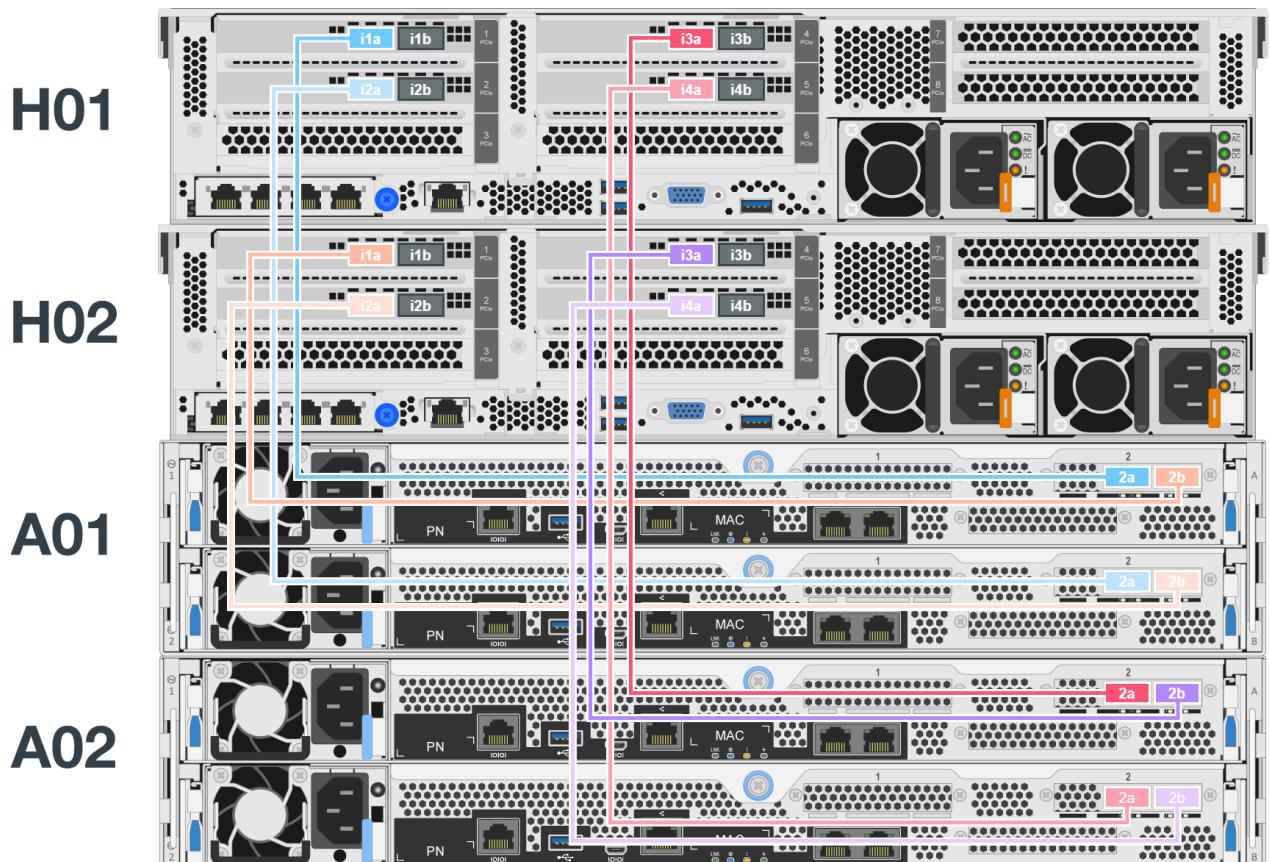
La configuración de la fuente de alimentación para los casos de uso de producción normalmente debe utilizar fuentes de alimentación redundantes.

3. Si es necesario, instale las unidades en cada uno de los nodos de bloque BeeGFS.
 - a. Si se va a utilizar el bloque de creación para ejecutar metadatos y servicios de almacenamiento de BeeGFS y unidades más pequeñas para volúmenes de metadatos, compruebe que estén ocupados en las ranuras de unidad más externas, como se muestra en la siguiente figura.
 - b. Para todas las configuraciones de bloques de construcción, si un compartimento de unidades no está completamente cargado, asegúrese de que se llena un mismo número de unidades en las ranuras 0–11 y 12–23 para obtener un rendimiento óptimo.

RAID 6 (8+2) Data



- Conecte los nodos de bloque y archivo con "[1M cables de cobre de conexión directa InfiniBand HDR 200GB](#)", de modo que coincidan con la topología que se muestra en la siguiente figura.





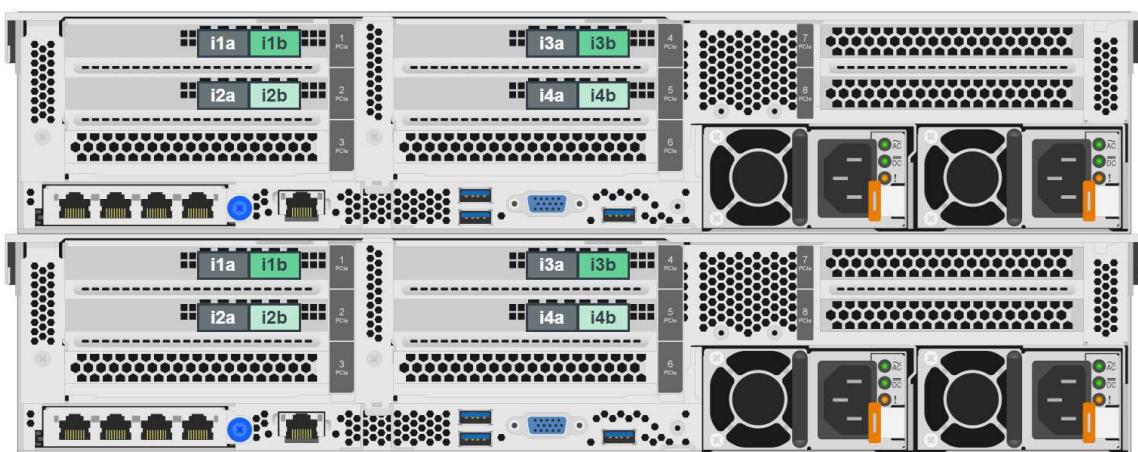
Los nodos de varios elementos básicos nunca están conectados directamente. Cada bloque se debe tratar como una unidad independiente y toda la comunicación entre los bloques de construcción se produce a través de conmutadores de red.

5. Conecte los puertos InfiniBand restantes del nodo de archivos al switch InfiniBand de la red de almacenamiento mediante el "["2m cables InfiniBand"](#)" switch de almacenamiento InfiniBand específico del switch de almacenamiento.

Cuando se utilizan cables divisores para conectar el switch de almacenamiento a nodos de archivos, un cable debe ramificarse del switch y conectarse a los puertos descritos en verde claro. Otro cable del divisor debe desviarse del interruptor y conectarse a los puertos señalados en verde oscuro.

Además, para las redes de almacenamiento con switches redundantes, los puertos descritos en verde claro deben conectarse a un switch, mientras que los puertos de verde oscuro deben conectarse a otro switch.

H01



H02

6. Según sea necesario, monte elementos básicos adicionales siguiendo las mismas directrices de cableado.



El número total de elementos básicos que se pueden poner en marcha en un único rack depende de la alimentación y la refrigeración disponibles en cada sitio.

Poner en marcha el software

Configure los nodos de archivo y los nodos de bloque

Aunque la mayoría de las tareas de configuración de software se automatizan a través de las colecciones Ansible proporcionadas por NetApp, debe configurar la red en la controladora de gestión de placa base (BMC) de cada servidor y configurar el puerto de gestión de cada controladora.

Configure los nodos de archivo

1. Configure la red en el controlador de administración de la placa base (BMC) de cada servidor.

Para obtener información sobre cómo configurar la red para los nodos de archivo SR665 V3 de Lenovo validados, consulte la "["Documentación de Lenovo ThinkSystem"](#)".



Un controlador de administración en placa base (BMC), conocido a veces como procesador de servicios, es el nombre genérico para la capacidad de administración fuera de banda integrada en varias plataformas de servidor que pueden proporcionar acceso remoto aunque el sistema operativo no esté instalado o sea accesible. Los proveedores suelen comercializar esta funcionalidad con su propia Marca. Por ejemplo, en el Lenovo SR665, el BMC se denomina *Lenovo XClarity Controller (XCC)*.

2. Configure los ajustes del sistema para obtener el máximo rendimiento.

Puede configurar los ajustes del sistema utilizando la configuración UEFI (anteriormente conocida como BIOS) o utilizando las API Redfish proporcionadas por muchos BMCs. La configuración del sistema varía según el modelo de servidor utilizado como nodo de archivo.

Para aprender a configurar los ajustes del sistema para los nodos de archivos Lenovo SR665 V3 validados, consulte "[Ajuste la configuración del sistema para obtener rendimiento](#)" .

3. Instale Red Hat Enterprise Linux (RHEL) 9.4 y configure el nombre de host y el puerto de red utilizados para administrar el sistema operativo, incluida la conectividad SSH desde el nodo de control Ansible.

No configure las IP en ninguno de los puertos InfiniBand en este momento.



Si bien no es estrictamente necesario, las secciones posteriores suponen que los nombres de host están numerados secuencialmente (como h1-HN) y hacen referencia a tareas que deben completarse en hosts pares versus impar.

4. Utilice Red Hat Subscription Manager para registrar y suscribir el sistema para permitir la instalación de los paquetes necesarios desde los repositorios oficiales de Red Hat y para limitar las actualizaciones a la versión compatible de Red Hat: `subscription-manager release --set=9.4`. Para obtener instrucciones, consulte "[Cómo registrar y suscribirse a un sistema RHEL](#)" y "[Cómo limitar las actualizaciones](#)".

5. Active el repositorio de Red Hat que contiene los paquetes necesarios para la alta disponibilidad.

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

6. Actualice todo el firmware de HCA a la versión recomendada en "[Requisitos tecnológicos](#)" Uso de la "[Actualice el firmware del adaptador de nodo de archivo](#)" guía.

Configure los nodos de bloque

Configure los nodos de bloque EF600 configurando el puerto de gestión en cada controladora.

1. Configure el puerto de gestión en cada controladora EF600.

Para obtener instrucciones sobre la configuración de puertos, vaya a la "[Centro de documentación de E-Series](#)".

2. De manera opcional, establezca el nombre de la cabina de almacenamiento para cada sistema.

Establecer un nombre puede facilitar la referencia a cada sistema en las secciones siguientes. Para obtener instrucciones sobre cómo configurar el nombre de la matriz, vaya a la "[Centro de documentación de E-Series](#)".



Si bien no es estrictamente necesario, los temas posteriores presuponen que los nombres de las cabinas de almacenamiento están numerados secuencialmente (como c1 - CN) y consulte los pasos que deben completarse en sistemas impar frente a sistemas numerados.

Ajuste la configuración del sistema del nodo de archivos para aumentar el rendimiento

Para maximizar el rendimiento, recomendamos configurar los ajustes del sistema en el modelo de servidor que utilice como nodos de archivos.

La configuración del sistema varía en función del modelo de servidor que utilice como nodo de archivos. En este tema se describe cómo configurar los valores del sistema para los nodos de archivo del servidor Lenovo ThinkSystem SR665 validados.

Utilice la interfaz UEFI para ajustar la configuración del sistema

El firmware del sistema del servidor Lenovo SR665 V3 contiene numerosos parámetros de ajuste que se pueden configurar a través de la interfaz UEFI. Estos parámetros de ajuste pueden afectar a todos los aspectos del funcionamiento del servidor y el rendimiento del mismo.

En **Configuración de UEFI > Configuración del sistema**, ajuste los siguientes ajustes del sistema:

Menú modo de funcionamiento

Ajuste del sistema	Cambiar a
Modo de funcionamiento	Personalizado
CTDP	Manual
Manual de cTDP	350
Límite de alimentación del paquete	Manual
Modo de eficiencia	Desactivar
Global-Estado-Control	Desactivar
Estados SOC P.	P0
DF C-Estados	Desactivar
Estado P.	Desactivar
Activación de la desconexión de memoria	Desactivar
Nodos NUMA por socket	NPS1

Menú dispositivos y puertos de E/S.

Ajuste del sistema	Cambiar a
IOMMU	Desactivar

Menú de encendido

Ajuste del sistema	Cambiar a
Freno de alimentación PCIe	Desactivar

Menú procesadores

Ajuste del sistema	Cambiar a
Control de estado C global	Desactivar
DF C-Estados	Desactivar
Modo SMT	Desactivar
CPPC	Desactivar

Utilice la API de Redfish para ajustar la configuración del sistema

Además de utilizar UEFI Setup, puede utilizar la API Redfish para cambiar la configuración del sistema.

```

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_ChooseOperatingMode": "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"
}
}
'

```

Para obtener información detallada sobre el esquema Redfish, consulte ["Sitio web de DMTF"](#).

Configure un nodo de control Ansible

Para configurar un nodo de control de Ansible, debe designar una máquina virtual o física con acceso de red a todos los nodos de archivos y bloques que se han puesto en marcha para la solución BeeGFS en NetApp.

Consulte la ["Requisitos técnicos"](#) para obtener una lista de versiones de paquete recomendadas. Los siguientes pasos fueron probados en Ubuntu 22,04. Para conocer los pasos específicos de su distribución de Linux preferida, consulte la ["Documentación de Ansible"](#).

1. Desde el nodo de control de Ansible, instale los siguientes paquetes de entorno virtual de Python y Python.

```

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

```

2. Crear un entorno virtual de Python.

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

3. Activar el entorno virtual.

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

4. Instale los paquetes de Python necesarios dentro del entorno virtual activado.

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

5. Instale la colección BeeGFS con Ansible Galaxy.

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

6. Compruebe que las versiones instaladas de Ansible, Python y la colección BeeGFS coinciden con el ["Requisitos técnicos"](#).

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

7. Configure SSH sin contraseña para permitir que Ansible acceda a los nodos de archivos BeeGFS remotos desde el nodo de control Ansible.

- En el nodo de control de Ansible, si es necesario, genere un par de claves públicas.

```
ssh-keygen
```

- Configure SSH sin contraseña en cada uno de los nodos de archivo.

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



No configure SSH sin contraseñas para los nodos de bloque. No se admite ni se requiere.

Cree el inventario de Ansible

Para definir la configuración de los nodos de archivos y bloques, debe crear un inventario de Ansible que represente el sistema de archivos BeeGFS que desea implementar. El inventario incluye hosts, grupos y variables que describen el sistema de archivos BeeGFS deseado.

Paso 1: Definir la configuración para todos los bloques de construcción

Defina la configuración que se aplica a todos los bloques de creación, independientemente del perfil de configuración que pueda aplicar a ellos individualmente.

Antes de empezar

- Seleccione un esquema de direcciones de subred para el despliegue. Debido a las ventajas que se muestran en "[arquitectura de software](#)", se recomienda utilizar un esquema de direcciones de subred única.

Pasos

1. En el nodo de control de Ansible, identifique un directorio que desea usar para almacenar los archivos del inventario y el libro de estrategia de Ansible.

A menos que se indique lo contrario, todos los archivos y directorios creados en este paso y los pasos siguientes se crean en relación con este directorio.

2. Cree los siguientes subdirectorios:

host_vars

group_vars

packages

3. Cree un subdirectorio para las contraseñas del clúster y asegure el archivo cifrándolo con Ansible Vault (consulte "[Cifrado de contenido con Ansible Vault](#)"):

a. Crear el subdirectorio group_vars/all.

b. En el group_vars/all directorio, cree un archivo de contraseñas con la etiqueta passwords.yml.

c. Rellene el passwords.yml file con los siguientes parámetros, sustituyendo todos los parámetros de nombre de usuario y contraseña según su configuración:

```

# 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. Ejecute `ansible-vault encrypt passwords.yml` y establezca una contraseña de almacenamiento cuando se le solicite.

Paso 2: Definir la configuración para nodos de archivos y bloques individuales

Defina la configuración que se aplica a los nodos de archivo individuales y a los nodos individuales de los bloques de creación.

1. Inferior `host_vars/`, Cree un archivo para cada nodo de archivo BeeGFS denominado `<HOSTNAME>.yml` Con el siguiente contenido, prestando especial atención a las notas relativas al contenido que se debe rellenar para los nombres de host y IP del clúster BeeGFS que terminan en números pares y pares.

Inicialmente, los nombres de la interfaz del nodo de archivos coinciden con los que se enumeran aquí (como ib0 o ibs1f0). Estos nombres personalizados se configuran en [Paso 4: Defina la configuración que debe aplicarse a todos los nodos de archivo](#).

```

ansible_host: "<MANAGEMENT_IP>"
eseries_ipoib_interfaces: # Used to configure BeeGFS cluster IP
addresses.
  - name: i1b
    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

```



Si ya ha implementado el clúster BeeGFS, debe detener el clúster antes de añadir o cambiar direcciones IP configuradas de forma estática, incluidas las IP y las IP del clúster utilizadas para NVMe/IB. Esto es necesario para que estos cambios entren en vigencia correctamente y no interrumpan las operaciones del clúster.

2. Inferior `host_vars/`, Cree un archivo para cada nodo de bloque BeeGFS denominado `<HOSTNAME>.yml` y rellene con el siguiente contenido.

Preste especial atención a las notas en relación con el contenido para llenar los nombres de las cabinas de almacenamiento que terminan en números impar frente a pares.

Para cada nodo de bloque, cree un archivo y especifique el `<MANAGEMENT_IP>` Para una de las dos controladoras (generalmente A).

```
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
```

Paso 3: Defina la configuración que debe aplicarse a todos los nodos de archivo y bloque

Puede definir la configuración común a un grupo de hosts en `group_vars` en un nombre de archivo que corresponde al grupo. Esto evita la repetición de una configuración compartida en varios lugares.

Acerca de esta tarea

Los hosts pueden estar en más de un grupo y, en tiempo de ejecución, Ansible elige qué variables aplican a un host determinado basándose en sus reglas de prioridad variable. (Para obtener más información sobre estas reglas, consulte la documentación de Ansible para "[Uso de variables](#)".)

Las asignaciones de hosts a grupos se definen en el archivo de inventario real de Ansible, que se crea hacia el final de este procedimiento.

Paso

En Ansible, se puede definir cualquier configuración que deseé aplicar a todos los hosts en un grupo llamado All. Cree el archivo `group_vars/all.yml` con el siguiente contenido:

```
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"
```

Paso 4: Defina la configuración que debe aplicarse a todos los nodos de archivo

La configuración compartida para los nodos de archivo se define en un grupo denominado `ha_cluster`. Los pasos de esta sección crean la configuración que se debe incluir en `group_vars/ha_cluster.yml` archivo.

Pasos

1. En la parte superior del archivo, defina los valores predeterminados, incluida la contraseña que se utilizará como sudo usuario en los nodos de archivo.

```
### 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"
```



Si el `ansible_ssh_user` ya está `root`, puede omitir y especificar la `--ask-become-pass` opción al ejecutar el `ansible_become_password` playbook.

2. Opcionalmente, configure un nombre para el clúster de alta disponibilidad (ha) y especifique un usuario para la comunicación dentro del clúster.

Si está modificando el esquema de direcciones IP privadas, también debe actualizar el valor predeterminado `beegfs_ha_mgmtd_floating_ip`. Esto debe coincidir con lo que configure más adelante para el grupo de recursos BeeGFS Management.

Especifique uno o más correos electrónicos que deben recibir alertas para eventos del clúster mediante `beegfs_ha_alert_email_list`.

```

### 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_verbose: 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

```



Aunque aparentemente redundante, `beegfs_ha_mgmtd_floating_ip` Es importante cuando escala el sistema de archivos BeeGFS más allá de un único clúster de alta disponibilidad. Los clústeres de alta disponibilidad posteriores se ponen en marcha sin un servicio de gestión de BeeGFS adicional y se señalan en el servicio de gestión proporcionado por el primer clúster.

- Configure un agente de cercado. (Para obtener más información, consulte ["Configurar la delimitación en un clúster de alta disponibilidad de Red Hat"](#).) En la siguiente salida se muestran ejemplos para configurar agentes de delimitación comunes. Elija una de estas opciones.

Para este paso, tenga en cuenta que:

- De forma predeterminada, la delimitación está activada, pero necesita configurar un elemento *agent* de cercado.
- La <HOSTNAME> especificado en la `pcmk_host_map` o `pcmk_host_list` Debe corresponder con el nombre de host del inventario de Ansible.
- No se admite la ejecución del clúster BeeGFS sin vallado, especialmente en producción. Esto se debe en gran medida a que los servicios BeeGFS, incluidas las dependencias de recursos como los dispositivos de bloque, comutan por error debido a un problema, no existe riesgo de acceso simultáneo por parte de varios nodos que provocan daños en el sistema de archivos u otro comportamiento inesperado o no deseado. Si es necesario desactivar el cercado, consulte las notas generales de la guía de inicio y ajuste del rol BeeGFS ha `beegfs_ha_cluster_crm_config_options["stonith-enabled"]` a falso in `ha_cluster.yml`.
- Hay varios dispositivos de cercado a nivel de nodo disponibles y el rol BeeGFS ha puede configurar cualquier agente de cercado disponible en el repositorio de paquetes de alta disponibilidad de Red Hat. Cuando sea posible, utilice un agente de esgrima que funcione a través del sistema de alimentación ininterrumpida (UPS) o de la unidad de distribución de alimentación en rack (rPDU). Debido a que algunos agentes de cercado, como el controlador de administración de la placa base (BMC) u otros dispositivos de apagado que están integrados en el servidor, puede que no respondan a la solicitud de cercado en determinados casos de fallo.

```

### 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\_availability\_clusters/assembly\_configuring-fencing-configuring-and-managing-high-availability-clusters.

```

4. Habilite el ajuste de rendimiento recomendado en el sistema operativo Linux.

Aunque muchos usuarios encuentran la configuración predeterminada para los parámetros de rendimiento por lo general funciona bien, de manera opcional, puede cambiar la configuración predeterminada para una carga de trabajo en particular. Como tal, estas recomendaciones se incluyen en el rol BeeGFS, pero no están habilitadas de forma predeterminada para garantizar que los usuarios conozcan el ajuste aplicado a su sistema de archivos.

Para habilitar el ajuste de rendimiento, especifique lo siguiente:

```
### Performance Configuration:  
beegfs_ha_enable_performance_tuning: True
```

5. (Opcional) puede ajustar los parámetros de ajuste del rendimiento en el sistema operativo Linux según sea necesario.

Para obtener una lista completa de los parámetros de ajuste disponibles que puede ajustar, consulte la sección Valores predeterminados de ajuste de rendimiento del rol BeeGFS HA en "[Sitio de E-Series BeeGFS GitHub](#)". Los valores por defecto se pueden sustituir para todos los nodos del cluster en este archivo o para el `host_vars` archivo de un nodo individual.

6. Para permitir una conectividad 200GB/HDR completa entre los nodos de bloques y archivos, utilice el paquete Administrador de subred abierta (OpenSM) de la distribución empresarial de estructuras abiertas de NVIDIA (MLNX_OFED). La versión MLNX_OFED de la lista "[requisitos del nodo de archivo](#)" incluye los paquetes OpenSM recomendados. Aunque la implementación mediante Ansible es compatible, primero debe instalar el controlador MLNX_OFED en todos los nodos de archivos.
 - a. Rellene los siguientes parámetros en `group_vars/ha_cluster.yml` (ajuste los paquetes según sea necesario):

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

7. Configure el `udev` Regla para garantizar la asignación coherente de identificadores de puerto InfiniBand lógicos a dispositivos PCIe subyacentes.

La `udev` La regla debe ser exclusiva de la topología PCIe de cada plataforma de servidor utilizada como nodo de archivo BeeGFS.

Utilice los siguientes valores para nodos de archivo verificados:

```

#### 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. (Opcional) Actualice el algoritmo de selección del objetivo de metadatos.

```

beegfs_ha_beegfs_meta_conf_ha_group_options:
    tuneTargetChooser: randomrobin

```



En las pruebas de verificación, `randomrobin` normalmente se utilizó para garantizar que los archivos de prueba se distribuyeron uniformemente en todos los destinos de almacenamiento de BeeGFS durante las pruebas de rendimiento (para obtener más información sobre pruebas de rendimiento, consulte el sitio de BeeGFS para ["Evaluación comparativa de un sistema BeeGFS"](#)). Con el uso en el mundo real, esto podría hacer que los blancos numerados más bajos se llenen más rápido que los blancos numerados más altos. Omitiendo `randomrobin` y sólo con el valor predeterminado `randomized` se ha demostrado que el valor proporciona un buen rendimiento mientras se siguen utilizando todos los objetivos disponibles.

Paso 5: Defina la configuración para el nodo de bloques común

La configuración compartida para los nodos de bloque se define en un grupo denominado `eseries_storage_systems`. Los pasos de esta sección crean la configuración que se debe incluir en `group_vars/ eseries_storage_systems.yml` archivo.

Pasos

1. Establezca la conexión de Ansible como local, proporcione la contraseña del sistema y especifique si deben verificarse los certificados SSL. (Normalmente, Ansible utiliza SSH para conectar a hosts gestionados; sin embargo, en el caso de los sistemas de almacenamiento E-Series de NetApp que se utilizan como nodos de bloques, los módulos usan la API REST para la comunicación.) En la parte superior del archivo, añada lo siguiente:

```
## 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. Para garantizar un rendimiento óptimo, instale las versiones enumeradas para los nodos de bloques en "[Requisitos técnicos](#)".

Descargue los archivos correspondientes de la "[Sitio de soporte de NetApp](#)". Puede actualizarlos manualmente o incluirlos en la packages/ directorio del nodo de control de Ansible y, a continuación, rellene los siguientes parámetros en `eseries_storage_systems.yml`. Para actualizar con Ansible:

```
# 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. Descargue e instale el firmware de la unidad más reciente disponible para las unidades instaladas en los nodos de bloque en el "[Sitio de soporte de NetApp](#)". Puede actualizarlos manualmente o incluirlos en packages/ el directorio del nodo de control de Ansible y, a continuación, llenar los siguientes parámetros en `eseries_storage_systems.yml` la actualización mediante Ansible:

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



Ajuste `eseries_drive_firmware_upgrade_drives_online` para `false`. Agiliza la actualización, pero no se debe realizar hasta después de que BeeGFS se haya puesto en marcha. Esto se debe a que esta configuración requiere detener todas las operaciones de I/o de las unidades antes de la actualización para evitar errores en las aplicaciones. Aunque realizar una actualización del firmware de la unidad en línea antes de configurar volúmenes es todavía rápida, se recomienda configurar siempre este valor en `true` para evitar problemas más adelante.

4. Para optimizar el rendimiento, realice los siguientes cambios en la configuración global:

```
# 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. Para garantizar un comportamiento y aprovisionamiento de volúmenes óptimos, especifique los siguientes parámetros:

```
# 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"
```



Valor especificado para `eseries_storage_pool_usable_drives`. Es específico de los nodos de bloques EF600 de NetApp y controla el orden en que se asignan las unidades a los nuevos grupos de volúmenes. Este pedido garantiza que la I/o de cada grupo se distribuya de forma uniforme en todos los canales de unidades del back-end.

Defina el inventario de Ansible para los bloques de creación de BeeGFS

Después de definir la estructura general de inventario de Ansible, defina la configuración de cada bloque de creación en el sistema de archivos BeeGFS.

Estas instrucciones de implementación muestran cómo instalar un sistema de archivos que consiste en un elemento básico que incluye servicios de gestión, metadatos y almacenamiento; un segundo elemento básico con servicios de metadatos y almacenamiento y un tercer elemento básico solo para el almacenamiento.

El objetivo de estos pasos es mostrar toda la gama de perfiles de configuración típicos que se pueden utilizar para configurar bloques de creación de BeeGFS de NetApp de modo que se cumplan los requisitos del sistema de archivos BeeGFS general.



En esta y en secciones posteriores, ajuste según sea necesario para generar el inventario que represente el sistema de archivos BeeGFS que desea implementar. En concreto, utilice los nombres de host de Ansible que representan cada nodo de bloque o archivo y el esquema de direccionamiento IP deseado para la red de almacenamiento a fin de garantizar que puede escalarse hasta el número de clientes y nodos de archivos BeeGFS.

Paso 1: Cree el archivo de inventario de Ansible

Pasos

1. Cree un nuevo `inventory.yml` archivo e inserte los siguientes parámetros, reemplazando los hosts en `eseries_storage_systems` según sea necesario, para representar los nodos de bloques en su puesta en marcha. Los nombres deben corresponder con el nombre utilizado para `host_vars/<FILENAME>.yml`.

```
# 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:
```

En las secciones siguientes, creará grupos de Ansible adicionales en `ha_cluster` Que representan los servicios BeeGFS que desea ejecutar en el clúster.

Paso 2: Configure el inventario para un elemento básico de gestión, metadatos y almacenamiento

El primer elemento básico del clúster o bloque básico debe incluir el servicio de gestión de BeeGFS junto con los servicios de metadatos y almacenamiento:

Pasos

1. Pulg `inventory.yml`, rellene los siguientes parámetros en `ha_cluster: children:`

```
# 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. Cree el archivo `group_vars/mgmt.yml` e incluya lo siguiente:

```

# mgmt - BeeGFS HA Management Resource Group
# OPTIONAL: Override default BeeGFS management configuration:
# beegfs_ha_beegfs_mgmtd_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. Inferior `group_vars/`, cree archivos para grupos de recursos `meta_01` por `meta_08` utilice la siguiente plantilla y, a continuación, rellene los valores de marcador de posición de cada servicio que haga referencia a la siguiente tabla:

```

# 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>

```



El tamaño del volumen se especifica como un porcentaje del pool de almacenamiento general (también denominado grupo de volúmenes). NetApp recomienda encarecidamente que deje cierta capacidad libre en cada pool para dejar espacio para el sobreaprovisionamiento de SSD (para obtener más información, consulte ["Introducción a la cabina EF600 de NetApp"](#)). El pool de almacenamiento, beegfs_m1_m2_m5_m6, también asigna el 1% de la capacidad del pool para el servicio de administración. Por lo tanto, para volúmenes de metadatos en el pool de almacenamiento, beegfs_m1_m2_m5_m6, Cuando se utilizan unidades de 1,92 TB o 3,84 TB, establezca este valor en 21.25; Para unidades de 7,65 TB, establezca este valor en 22.25; Y para las unidades de 15,3 TB, establezca este valor en 23.75.

Nombre de archivo	Puerto	IP flotantes	Zona NUMA	Nodo de bloques	Del banco de almacenamiento	Controladora propietaria
meta_01.yml	8015	i1b: 100.127.101. 1/16 i2b:100.127.1 02.1/16	0	netapp_01	beegfs_m1_ m2_m5_m6	A.
meta_02.yml	8025	i2b: 100.127.102. 2/16 i1b:100.127.1 01.2/16	0	netapp_01	beegfs_m1_ m2_m5_m6	B

Nombre de archivo	Puerto	IP flotantes	Zona NUMA	Nodo de bloques	Del banco de almacenamiento	Controladora propietaria
meta_03.yml	8035	i3b: 100.127.101. 3/16 i4b:100.127.1 02.3/16	1	netapp_02	beegfs_m3_ m4_m7_m8	A.
meta_04.yml	8045	i4b: 100.127.102. 4/16 i3b:100.127.1 01.4/16	1	netapp_02	beegfs_m3_ m4_m7_m8	B
meta_05.yml	8055	i1b: 100.127.101. 5/16 i2b:100.127.1 02.5/16	0	netapp_01	beegfs_m1_ m2_m5_m6	A.
meta_06.yml	8065	i2b: 100.127.102. 6/16 i1b:100.127.1 01.6/16	0	netapp_01	beegfs_m1_ m2_m5_m6	B
meta_07.yml	8075	i3b: 100.127.101. 7/16 i4b:100.127.1 02.7/16	1	netapp_02	beegfs_m3_ m4_m7_m8	A.
meta_08.yml	8085	i4b: 100.127.102. 8/16 i3b:100.127.1 01.8/16	1	netapp_02	beegfs_m3_ m4_m7_m8	B

4. Inferior `group_vars/`, cree archivos para grupos de recursos `stor_01` por `stor_08` utilizando la siguiente plantilla y, a continuación, rellene los valores de marcador de posición para cada servicio que haga referencia al ejemplo:

```

# 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>

```



Para ver el tamaño correcto de uso, consulte ["Se recomendaron porcentajes de sobreaprovisionamiento del pool de almacenamiento"](#).

Nombre de archivo	Puerto	IP flotantes	Zona NUMA	Nodo de bloques	Del banco de almacenamiento	Controladora propietaria
stor_01.yml	8013	i1b: 100.127.103. 1/16 i2b:100.127.1 04.1/16	0	netapp_01	beegfs_s1_s2	A.
stor_02.yml	8023	i2b: 100.127.104. 2/16 i1b:100.127.1 03.2/16	0	netapp_01	beegfs_s1_s2	B
stor_03.yml	8033	i3b: 100.127.103. 3/16 i4b:100.127.1 04.3/16	1	netapp_02	beegfs_s3_s4	A.

Nombre de archivo	Puerto	IP flotantes	Zona NUMA	Nodo de bloques	Del banco de almacenamiento	Controladora propietaria
stor_04.yml	8043	i4b: 100.127.104. 4/16 i3b:100.127.1 03.4/16	1	netapp_02	beegfs_s3_s4	B
stor_05.yml	8053	i1b: 100.127.103. 5/16 i2b:100.127.1 04.5/16	0	netapp_01	beegfs_s5_s6	A.
stor_06.yml	8063	i2b: 100.127.104. 6/16 i1b:100.127.1 03.6/16	0	netapp_01	beegfs_s5_s6	B
stor_07.yml	8073	i3b: 100.127.103. 7/16 i4b:100.127.1 04.7/16	1	netapp_02	beegfs_s7_s8	A.
stor_08.yml	8083	i4b: 100.127.104. 8/16 i3b:100.127.1 03.8/16	1	netapp_02	beegfs_s7_s8	B

Paso 3: Configure el inventario para un bloque básico de metadatos + almacenamiento

Estos pasos describen cómo configurar un inventario de Ansible para un elemento básico de metadatos BeeGFS + almacenamiento.

Pasos

1. Pulg `inventory.yml`, rellene los siguientes parámetros bajo la configuración existente:

```

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:
    beegfs_03:

```

2. Inferior `group_vars/`, cree archivos para grupos de recursos `meta_09` por `meta_16` utilizando la siguiente plantilla y, a continuación, rellene los valores de marcador de posición para cada servicio que haga referencia al ejemplo:

```

# 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>

```



Para ver el tamaño correcto de uso, consulte "["Se recomendaron porcentajes de sobreaprovisionamiento del pool de almacenamiento"](#)".

Nombre de archivo	Puerto	IP flotantes	Zona NUMA	Nodo de bloques	Del banco de almacenamiento	Controladora propietaria
meta_09.yml	8015	i1b: 100.127.101. 9/16 i2b:100.127.1 02.9/16	0	netapp_03	beegfs_m9_ m10_m13_m 14	A.

Nombre de archivo	Puerto	IP flotantes	Zona NUMA	Nodo de bloques	Del banco de almacenamiento	Controladora propietaria
meta_10.yml	8025	i2b: 100.127.102. 10/16 i1b:100.127.1 01.10/16	0	netapp_03	beegfs_m9_ m10_m13_m 14	B
meta_11.yml	8035	i3b: 100.127.101. 11/16 i4b:100.127.1 02.11/16	1	netapp_04	beegfs_m11_ m12_m15_m 16	A.
meta_12.yml	8045	i4b: 100.127.102. 12/16 i3b:100.127.1 01.12/16	1	netapp_04	beegfs_m11_ m12_m15_m 16	B
meta_13.yml	8055	i1b: 100.127.101. 13/16 i2b:100.127.1 02.13/16	0	netapp_03	beegfs_m9_ m10_m13_m 14	A.
meta_14.yml	8065	i2b: 100.127.102. 14/16 i1b:100.127.1 01.14/16	0	netapp_03	beegfs_m9_ m10_m13_m 14	B
meta_15.yml	8075	i3b: 100.127.101. 15/16 i4b:100.127.1 02.15/16	1	netapp_04	beegfs_m11_ m12_m15_m 16	A.
meta_16.yml	8085	i4b: 100.127.102. 16/16 i3b:100.127.1 01.16/16	1	netapp_04	beegfs_m11_ m12_m15_m 16	B

3. Inferior `group_vars/`, crear archivos para grupos de recursos `stor_09` por `stor_16` utilizando la siguiente plantilla y, a continuación, rellene los valores de marcador de posición para cada servicio que haga referencia al ejemplo:

```

# 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>

```



Para conocer el tamaño correcto a utilizar, consulte "["Se recomendaron porcentajes de sobreaprovisionamiento del pool de almacenamiento"](#) ..

Nombre de archivo	Puerto	IP flotantes	Zona NUMA	Nodo de bloques	Del banco de almacenamiento	Controladora propietaria
stor_09.yml	8013	i1b: 100.127.103. 9/16 i2b:100.127.1 04.9/16	0	netapp_03	beegfs_s9_s1 0	A.
stor_10.yml	8023	i2b: 100.127.104. 10/16 i1b:100.127.1 03.10/16	0	netapp_03	beegfs_s9_s1 0	B
stor_11.yml	8033	i3b: 100.127.103. 11/16 i4b:100.127.1 04.11/16	1	netapp_04	beegfs_s11_s 12	A.

Nombre de archivo	Puerto	IP flotantes	Zona NUMA	Nodo de bloques	Del banco de almacenamiento	Controladora propietaria
stor_12.yml	8043	i4b: 100.127.104. 12/16 i3b:100.127.1 03.12/16	1	netapp_04	beegfs_s11_s 12	B
stor_13.yml	8053	i1b: 100.127.103. 13/16 i2b:100.127.1 04.13/16	0	netapp_03	beegfs_s13_s 14	A.
stor_14.yml	8063	i2b: 100.127.104. 14/16 i1b:100.127.1 03.14/16	0	netapp_03	beegfs_s13_s 14	B
stor_15.yml	8073	i3b: 100.127.103. 15/16 i4b:100.127.1 04.15/16	1	netapp_04	beegfs_s15_s 16	A.
stor_16.yml	8083	i4b: 100.127.104. 16/16 i3b:100.127.1 03.16/16	1	netapp_04	beegfs_s15_s 16	B

Paso 4: Configure el inventario para un elemento básico de solo almacenamiento

Estos pasos describen cómo configurar un inventario de Ansible para un elemento básico solo de almacenamiento de BeeGFS. La principal diferencia entre configurar una configuración para un almacenamiento y metadatos frente a un elemento básico solo de almacenamiento es la omisión de todos los grupos de recursos de metadatos y las cambios `criteria_drive_count` de 10 a 12 por cada pool de almacenamiento.

Pasos

1. Pulga `inventory.yml`, rellene los siguientes parámetros bajo la configuración existente:

```

# 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. Inferior `group_vars/`, cree archivos para grupos de recursos `stor_17` por `stor_24` utilizando la siguiente plantilla y, a continuación, rellene los valores de marcador de posición para cada servicio que haga referencia al ejemplo:

```

# 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>

```



Para conocer el tamaño correcto a utilizar, consulte ["Se recomendaron porcentajes de sobreaprovisionamiento del pool de almacenamiento"](#).

Nombre de archivo	Puerto	IP flotantes	Zona NUMA	Nodo de bloques	Del banco de almacenamiento	Controladora propietaria
stor_17.yml	8013	i1b: 100.127.103. 17/16 i2b:100.127.1 04.17/16	0	netapp_05	beegfs_s17_s 18	A.
stor_18.yml	8023	i2b: 100.127.104. 18/16 i1b:100.127.1 03.18/16	0	netapp_05	beegfs_s17_s 18	B
stor_19.yml	8033	i3b: 100.127.103. 19/16 i4b:100.127.1 04.19/16	1	netapp_06	beegfs_s19_s 20	A.

Nombre de archivo	Puerto	IP flotantes	Zona NUMA	Nodo de bloques	Del banco de almacenamiento	Controladora propietaria
stor_20.yml	8043	i4b: 100.127.104. 20/16 i3b:100.127.1 03.20/16	1	netapp_06	beegfs_s19_s 20	B
stor_21.yml	8053	i1b: 100.127.103. 21/16 i2b:100.127.1 04.21/16	0	netapp_05	beegfs_s21_s 22	A.
stor_22.yml	8063	i2b: 100.127.104. 22/16 i1b:100.127.1 03.22/16	0	netapp_05	beegfs_s21_s 22	B
stor_23.yml	8073	i3b: 100.127.103. 23/16 i4b:100.127.1 04.23/16	1	netapp_06	beegfs_s23_s 24	A.
stor_24.yml	8083	i4b: 100.127.104. 24/16 i3b:100.127.1 03.24/16	1	netapp_06	beegfs_s23_s 24	B

Ponga en marcha BeeGFS

La puesta en marcha y gestión de la configuración implica la ejecución de uno o más libros de estrategia que contengan las tareas que Ansible necesita para ejecutar y llevar el sistema general al estado deseado.

Aunque todas las tareas se pueden incluir en un único libro de aplicaciones, en sistemas complejos, su gestión se torna difícil. Ansible permite crear y distribuir roles como una forma de empaquetar libros de estrategia reutilizables y contenido relacionado (por ejemplo: Variables predeterminadas, tareas y controladores). Para obtener más información, consulte la documentación de Ansible para "[Funciones](#)".

A menudo, los roles se distribuyen como parte de una colección Ansible que contiene roles y módulos relacionados. De este modo, estos libros de estrategia importan principalmente varias funciones distribuidas en las distintas colecciones de Ansible E-Series de NetApp.



Actualmente, se necesitan al menos dos elementos básicos (cuatro nodos de archivo) para implementar BeeGFS, a menos que se configure un dispositivo de quórum independiente como tiebreaker para mitigar cualquier problema al establecer quórum con un clúster de dos nodos.

Pasos

1. Cree un nuevo playbook.yml file e incluya lo siguiente:

```
# 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

```



este libro de estrategia ejecuta algunos pre_tasks Con ese fin, verifique que Python 3 esté instalado en los nodos de archivos y compruebe que las etiquetas de Ansible proporcionadas sean compatibles.

2. Utilice la ansible-playbook Comando con los archivos de inventario y libro de estrategia cuando esté listo para implementar BeeGFS.

La implementación se ejecutará todo `pre_tasks` Y, a continuación, solicite la confirmación del usuario antes de continuar con el despliegue real de BeeGFS.

Ejecute el siguiente comando, ajustando el número de horquillas según sea necesario (consulte la nota siguiente):

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



Especialmente para implementaciones más grandes, `forks` se recomienda sobrescribir el número predeterminado de forks (5) que utiliza el parámetro para aumentar el número de hosts que Ansible configura en paralelo. (Para obtener más información, consulte ["Control de la ejecución del libro de estrategia"](#).) El valor máximo depende de la potencia de procesamiento disponible en el nodo de control de Ansible. El ejemplo anterior de 20 se ejecutó en un nodo de control de Ansible virtual con 4 CPU (CPU Intel® Xeon® Gold 6146 a 3,20 GHz).

Según el tamaño de la puesta en marcha y el rendimiento de la red entre el nodo de control de Ansible y los nodos de archivo y bloque de BeeGFS, el tiempo de puesta en marcha puede variar.

Configurar clientes BeeGFS

Debe instalar y configurar el cliente BeeGFS en cualquier host que necesite acceder al sistema de archivos BeeGFS, como nodos de computación o GPU. Para esta tarea, puede usar Ansible y la colección BeeGFS.

Pasos

1. Si es necesario, configure SSH sin contraseñas desde el nodo de control de Ansible a cada uno de los hosts que desea configurar como clientes BeeGFS:

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

2. Inferior `host_vars/`, Cree un archivo para cada cliente BeeGFS denominado `<HOSTNAME>.yml` con el siguiente contenido, rellene el texto del marcador de posición con la información correcta para su entorno:

```
# 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>
```



Si se implementa con un esquema de direcciones de dos subredes, se deben configurar dos interfaces InfiniBand en cada cliente, una en cada una de las dos subredes IPoIB de almacenamiento. Si se utilizan las subredes de ejemplo y los rangos recomendados para cada servicio BeeGFS enumerados aquí, los clientes deben tener una interfaz configurada en el rango de 100.127.1.0 hasta 100.127.99.255 y la otra en 100.128.1.0 hasta 100.128.99.255.

3. Cree un archivo nuevo `client_inventory.yml` y, a continuación, rellene los siguientes parámetros en la parte superior:

```
# 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.
```



No almacene contraseñas en texto sin formato. En su lugar, use el almacén de Ansible (consulte la documentación de Ansible para ["Cifrado de contenido con Ansible Vault"](#)) o utilice la --ask-become-pass al ejecutar el libro de estrategia.

4. En la `client_inventory.yml` File, enumera todos los hosts que deben configurarse como clientes BeeGFS en `beegfs_clients`. Agrupe y, a continuación, especifique cualquier configuración adicional necesaria para crear el módulo de kernel de cliente BeeGFS.

```

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.

```



Cuando utilice los controladores OFED de NVIDIA, asegúrese de que `beegfs_client_ofed_include_path` apunte a la ruta correcta de inclusión de encabezado para su instalación de Linux. Para obtener más información, consulte la documentación de BeeGFS para "[Compatibilidad con RDMA](#)".

5. En la `client_inventory.yml` File, enumera los sistemas de archivos BeeGFS que desea montar en la parte inferior de cualquier definido previamente `vars`.

```

beegfs_client_mounts:
    - sysMgmtdHost: 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

```



La `beegfs_client_config` representa la configuración que se ha probado. Consulte la documentación incluida con `netapp_eseries.beegfs` colecciones `beegfs_client` función para una visión general completa de todas las opciones. Esto incluye detalles sobre el montaje de varios sistemas de archivos BeeGFS o el montaje del mismo sistema de archivos BeeGFS varias veces.

6. Cree un nuevo `client_playbook.yml` rellene los siguientes parámetros:

```

# 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

```



Omitir la importación de `netapp_eseries.host` recopilación y. `ipoib` Rol si ya ha instalado los controladores IB/RDMA necesarios y ha configurado las IP en las interfaces IPoIB adecuadas.

7. Para instalar y crear el cliente y montar BeeGFS, ejecute el siguiente comando:

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

8. Antes de poner el sistema de archivos BeeGFS en producción, **recomendamos encarecidamente** que inicie sesión en cualquier cliente y ejecute `beegfs-fsck --checkfs` para garantizar que se pueda acceder a todos los nodos y no se notifican problemas.

Escalabilidad más allá de cinco elementos básicos

Puede configurar Pacemaker y Corosync para escalar más allá de cinco bloques de construcción (10 nodos de archivo). Sin embargo, hay inconvenientes para los grandes grupos, y finalmente Pacemaker y Corosync imponen un máximo de 32 nodos.

NetApp solo ha probado clústeres de alta disponibilidad de BeeGFS para un máximo de 10 nodos; no se recomienda ni admite el escalado de clústeres individuales por encima de este límite. No obstante, los sistemas de archivos BeeGFS aún deben escalar más allá de los 10 nodos, lo cual en BeeGFS es la solución de NetApp.

Al poner en marcha varios clústeres de alta disponibilidad que contienen un subconjunto de los bloques de creación en cada sistema de archivos, puede escalar el sistema de archivos BeeGFS general de forma independiente de los límites recomendados o físicos en los mecanismos de agrupación en clústeres de alta disponibilidad subyacentes. En este caso, haga lo siguiente:

- Cree un nuevo inventario de Ansible que represente los clústeres de alta disponibilidad adicionales y, a continuación, omita la configuración de otro servicio de gestión. En su lugar, apunte la `beegfs_ha_mgmt_ip_floating_ip` variable en cada clúster adicional `ha_cluster.yml` Al IP del primer servicio de gestión de BeeGFS.

- Cuando agregue clústeres de alta disponibilidad adicionales al mismo sistema de archivos, asegúrese de lo siguiente:
 - Los ID del nodo BeeGFS son únicos.
 - Los nombres de archivo correspondientes a cada servicio en `group_vars` es único en todos los clústeres.
 - Las direcciones IP del cliente y del servidor BeeGFS son únicas en todos los clústeres.
 - El primer clúster de alta disponibilidad que contiene el servicio de gestión de BeeGFS se está ejecutando antes de intentar implementar o actualizar clústeres adicionales.
- Mantener inventarios para cada clúster ha por separado en su propio árbol de directorios.

Si se intenta mezclar los archivos de inventario para varios clústeres en un árbol de directorios, se podrían generar problemas en el modo en que el rol de ha de BeeGFS agrega la configuración aplicada a un clúster concreto.

 No es necesario que cada clúster de alta disponibilidad escale hasta cinco elementos básicos antes de crear uno nuevo. En muchos casos, utilizar menos bloques básicos por clúster resulta más fácil de gestionar. Uno de los métodos consiste en configurar los elementos básicos en cada rack como un clúster de alta disponibilidad.

Se recomendaron porcentajes de sobreaprovisionamiento del pool de almacenamiento

Si sigue los cuatro volúmenes estándar por configuración de pool de almacenamiento para bloques básicos de segunda generación, consulte la siguiente tabla.

Esta tabla recomienda porcentajes para usar como el tamaño del volumen en el `eseries_storage_pool_configuration` Para cada metadatos o destino de almacenamiento de BeeGFS:

Tamaño de la unidad	Tamaño
1,92 TB	18
3,84 TB	21.5
7,68 TB	22.5
15,3 TB	24

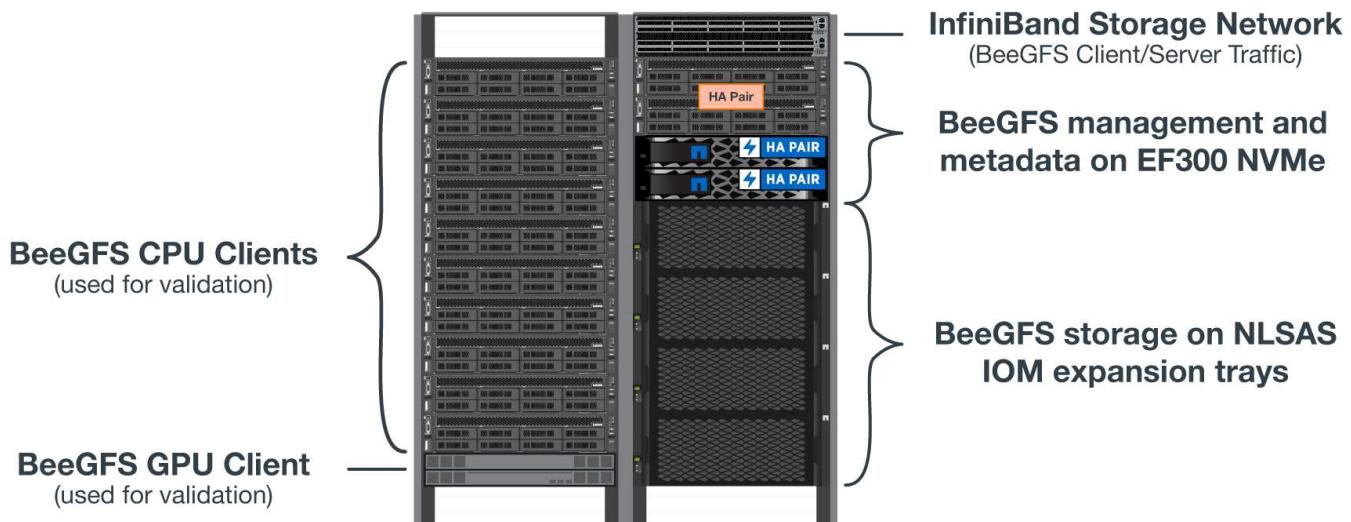
 Las directrices anteriores no se aplican al pool de almacenamiento que contiene el servicio de gestión, lo que debería reducir sus tamaños en un 25% para asignar el 1% del pool de almacenamiento a los datos de gestión.

Para entender cómo se determinaron estos valores, consulte "[TR-4800: Apéndice A: Aspectos sobre la resistencia de SSD y el sobreaprovisionamiento](#)".

Elemento básico de gran capacidad

La guía de implementación de la solución BeeGFS estándar describe procedimientos y recomendaciones para requisitos de alto rendimiento en las cargas de trabajo. Los

clientes que busquen cumplir requisitos de alta capacidad deben observar las variaciones en la implementación y las recomendaciones que se describen aquí.



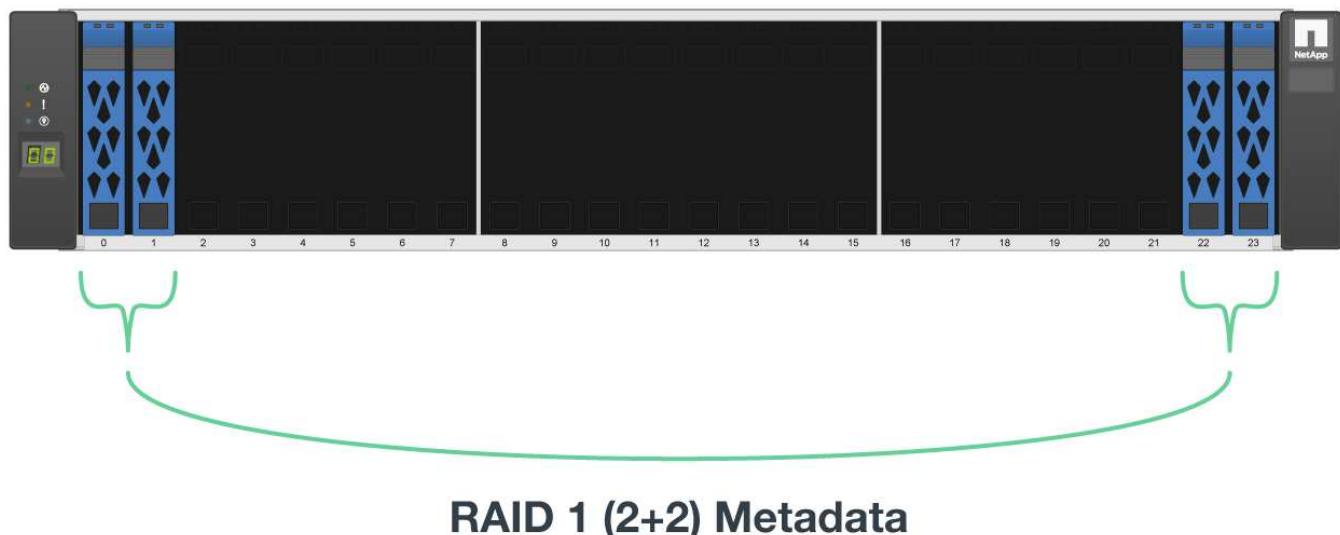
Controladoras

Para los elementos básicos de gran capacidad, las controladoras EF600 deben sustituirse por controladoras EF300, cada uno con una Cascade HIC instalada para la ampliación SAS. Cada nodo de bloque tendrá un número mínimo de SSD NVMe en el compartimento de la cabina para el almacenamiento de metadatos BeeGFS y se adjuntará a bandejas de expansión completas con HDD NL-SAS para volúmenes de almacenamiento BeeGFS.

La configuración del nodo de archivo al nodo de bloque sigue siendo la misma.

Ubicación de la unidad

Se requiere un mínimo de 4 SSD NVMe en cada nodo de bloque para el almacenamiento de metadatos BeeGFS. Estas unidades deben colocarse en las ranuras más externas de la carcasa.



Bandejas de expansión

El elemento básico de gran capacidad se puede ajustar con 1-7, 60 bandejas de expansión por cabina de almacenamiento.

Para obtener instrucciones para conectar cada bandeja de expansión, "[Consulte cableado EF300 para las bandejas de unidades](#)".

Información de copyright

Copyright © 2026 NetApp, Inc. Todos los derechos reservados. Impreso en EE. UU. No se puede reproducir este documento protegido por copyright ni parte del mismo de ninguna forma ni por ningún medio (gráfico, electrónico o mecánico, incluidas fotocopias, grabaciones o almacenamiento en un sistema de recuperación electrónico) sin la autorización previa y por escrito del propietario del copyright.

El software derivado del material de NetApp con copyright está sujeto a la siguiente licencia y exención de responsabilidad:

ESTE SOFTWARE LO PROPORCIONA NETAPP «TAL CUAL» Y SIN NINGUNA GARANTÍA EXPRESA O IMPLÍCITA, INCLUYENDO, SIN LIMITAR, LAS GARANTÍAS IMPLÍCITAS DE COMERCIALIZACIÓN O IDONEIDAD PARA UN FIN CONCRETO, CUYA RESPONSABILIDAD QUEDA EXIMIDA POR EL PRESENTE DOCUMENTO. EN NINGÚN CASO NETAPP SERÁ RESPONSABLE DE NINGÚN DAÑO DIRECTO, INDIRECTO, ESPECIAL, EJEMPLAR O RESULTANTE (INCLUYENDO, ENTRE OTROS, LA OBTENCIÓN DE BIENES O SERVICIOS SUSTITUTIVOS, PÉRDIDA DE USO, DE DATOS O DE BENEFICIOS, O INTERRUPCIÓN DE LA ACTIVIDAD EMPRESARIAL) CUALQUIERA SEA EL MODO EN EL QUE SE PRODUJERON Y LA TEORÍA DE RESPONSABILIDAD QUE SE APLIQUE, YA SEA EN CONTRATO, RESPONSABILIDAD OBJETIVA O AGRAVIO (INCLUIDA LA NEGLIGENCIA U OTRO TIPO), QUE SURJAN DE ALGÚN MODO DEL USO DE ESTE SOFTWARE, INCLUSO SI HUBIEREN SIDO ADVERTIDOS DE LA POSIBILIDAD DE TALES DAÑOS.

NetApp se reserva el derecho de modificar cualquiera de los productos aquí descritos en cualquier momento y sin aviso previo. NetApp no asume ningún tipo de responsabilidad que surja del uso de los productos aquí descritos, excepto aquello expresamente acordado por escrito por parte de NetApp. El uso o adquisición de este producto no lleva implícita ninguna licencia con derechos de patente, de marcas comerciales o cualquier otro derecho de propiedad intelectual de NetApp.

Es posible que el producto que se describe en este manual esté protegido por una o más patentes de EE. UU., patentes extranjeras o solicitudes pendientes.

LEYENDA DE DERECHOS LIMITADOS: el uso, la copia o la divulgación por parte del gobierno están sujetos a las restricciones establecidas en el subpárrafo (b)(3) de los derechos de datos técnicos y productos no comerciales de DFARS 252.227-7013 (FEB de 2014) y FAR 52.227-19 (DIC de 2007).

Los datos aquí contenidos pertenecen a un producto comercial o servicio comercial (como se define en FAR 2.101) y son propiedad de NetApp, Inc. Todos los datos técnicos y el software informático de NetApp que se proporcionan en este Acuerdo tienen una naturaleza comercial y se han desarrollado exclusivamente con fondos privados. El Gobierno de EE. UU. tiene una licencia limitada, irrevocable, no exclusiva, no transferible, no sublicenciable y de alcance mundial para utilizar los Datos en relación con el contrato del Gobierno de los Estados Unidos bajo el cual se proporcionaron los Datos. Excepto que aquí se disponga lo contrario, los Datos no se pueden utilizar, desvelar, reproducir, modificar, interpretar o mostrar sin la previa aprobación por escrito de NetApp, Inc. Los derechos de licencia del Gobierno de los Estados Unidos de América y su Departamento de Defensa se limitan a los derechos identificados en la cláusula 252.227-7015(b) de la sección DFARS (FEB de 2014).

Información de la marca comercial

NETAPP, el logotipo de NETAPP y las marcas que constan en <http://www.netapp.com/TM> son marcas comerciales de NetApp, Inc. El resto de nombres de empresa y de producto pueden ser marcas comerciales de sus respectivos propietarios.