BeeGFS en NetApp con almacenamiento E-Series

BeeGFS on NetApp with E-Series Storage

NetApp
October 22, 2024

Tabla de contenidos

BeeGFS en NetApp con almacenamiento E-Series

Manos a la obra

Lo que se incluye en este sitio

En este sitio se documenta cómo poner en marcha y gestionar BeeGFS en NetApp mediante arquitecturas verificadas de NetApp (NVA) y arquitecturas personalizadas. Los diseños de NVA se han probado exhaustivamente y proporcionan a los clientes configuraciones de referencia y directrices de tamaño para minimizar los riesgos de la puesta en marcha y acelerar el plazo de comercialización. NetApp también admite arquitecturas BeeGFS personalizadas que se ejecutan en hardware de NetApp, lo que ofrece a los clientes y partners flexibilidad para diseñar sistemas de archivos que cumplan una amplia gama de requisitos. Ambos métodos aprovechan Ansible para la puesta en marcha, lo que proporciona un enfoque similar al de los dispositivos para gestionar BeeGFS a cualquier escala en una variedad flexible de hardware.

Términos y conceptos

Los siguientes términos y conceptos se aplican a BeeGFS en la solución de NetApp.

Consulta la "Administre clústeres BeeGFS" sección para obtener información adicional sobre los términos y conceptos específicos para la interacción con clústeres de alta disponibilidad (HA) de BeeGFS.

Duración	Descripción
IA	Inteligencia artificial.
Inventario de Ansible	Estructura de directorio que contiene archivos YAML que se utilizan para describir el clúster de ha de BeeGFS deseado.
BMC	Controlador de administración de la placa base. En ocasiones se conoce como procesador de servicio.
nodos de bloques	Sistemas de almacenamiento.
clientes	Nodos del clúster HPC que ejecuta aplicaciones que deben utilizar el sistema de archivos. En ocasiones, también se denomina nodos de computación o GPU.
DL	Aprendizaje profundo.
nodos de archivos	Servidores de archivos BeeGFS.
HA	Alta disponibilidad.
HIC	Tarjeta de interfaz del host.

Duración	Descripción
HPC	Informática de alto rendimiento.
Cargas de trabajo de tipo HPC	Las cargas de trabajo de estilo HPC suelen caracterizarse por la necesidad de múltiples nodos de computación o GPU que necesiten acceder al mismo conjunto de datos en paralelo para facilitar una tarea de entrenamiento o computación distribuida. Estos conjuntos de datos a menudo constan de archivos de gran tamaño que se deben dividir entre varios nodos de almacenamiento físico para eliminar los cuellos de botella de hardware tradicionales, lo que evitaría el acceso simultáneo a un único archivo.
ML	Aprendizaje automático.
NLP	Procesamiento de Lenguaje Natural.
NLU	Comprensión del lenguaje natural.
ARQUITECTURA VALIDADA DE NETAPP	El programa NetApp Verified Architecture (NVA) proporciona configuraciones de referencia y directrices de tamaño para cargas de trabajo específicas y casos prácticos. Estas soluciones han sido probadas exhaustivamente y están diseñadas para minimizar los riesgos de la puesta en marcha y acelerar el plazo de comercialización.
red de almacenamiento/red de cliente	Red utilizada por los clientes para comunicarse con el sistema de archivos BeeGFS. A menudo, se trata de la misma red que se utiliza para la interfaz de paso de mensajes (MPI) en paralelo y otra comunicación de aplicaciones entre nodos de clúster HPC.

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".

Generaciones de diseño

BeeGFS para la solución de NetApp se encuentra en su segundo diseño generacional.

La primera y la segunda generación incluyen una arquitectura base que incorpora un sistema de archivos BeeGFS y un sistema de almacenamiento NVMe EF600. Sin embargo, la segunda generación se basa en la primera en incluir estas ventajas adicionales:

- Duplique el rendimiento y la capacidad al añadir solo 2U de espacio en rack
- Alta disponibilidad (ha) basada en un diseño de hardware de dos niveles y disco compartido
- Arquitectura diseñada para los sistemas DGX SuperPOD A100, H100, H200 y B200 de NVIDIA, que se validó previamente en un clúster de aceptación dedicado de NVIDIA. Obtenga más información sobre DGX SuperPOD de NVIDIA con NetApp en "guía de diseño"el .

Segundo diseño generacional

La segunda generación de BeeGFS en NetApp ha sido optimizada para satisfacer los requisitos de rendimiento de las cargas de trabajo más exigentes, como la computación de alto rendimiento (HPC), el aprendizaje automático (ML), el aprendizaje profundo (DL) y otras técnicas de inteligencia artificial (IA). Al incorporar una arquitectura de alta disponibilidad de discos compartidos, este diseño garantiza la durabilidad y la disponibilidad de los datos, lo que lo convierte en el ideal para empresas y otras organizaciones que no pueden permitirse tiempos de inactividad o pérdida de datos. El diseño de segunda generación incluye componentes como servidores PCIe Gen5 y soporte para conmutadores InfiniBand NVIDIA® Quantum™ QM9700 400GB/s. Esta solución no solo ha sido verificada por NetApp, sino que también ha superado la cualificación externa como opción de almacenamiento para el SuperPOD de NVIDIA DGX™ A100, con una certificación ampliada para los sistemas DGX SuperPOD H100, H200 y B200.

Primer diseño generacional

La primera generación de BeeGFS en NetApp se diseñó para cargas de trabajo de aprendizaje automático (ML) e inteligencia artificial (IA) utilizando sistemas de almacenamiento NVMe EF600 de NetApp, el sistema de archivos paralelo BeeGFS, los sistemas DGX™ A100 de NVIDIA y los switches IB NVIDIA® Mellanox® Quantum™ QM8700 200GB Gb/s. Este diseño también ofrece InfiniBand (IB) de 200GB Gb/s para la estructura de interconexión de clústeres de almacenamiento y de computación para proporcionar una arquitectura basada en IB completa para cargas de trabajo de alto rendimiento.

Para obtener más información sobre la primera generación, consulte "IA EF-Series de NetApp con sistemas NVIDIA DGX A100 y BeeGFS".

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.

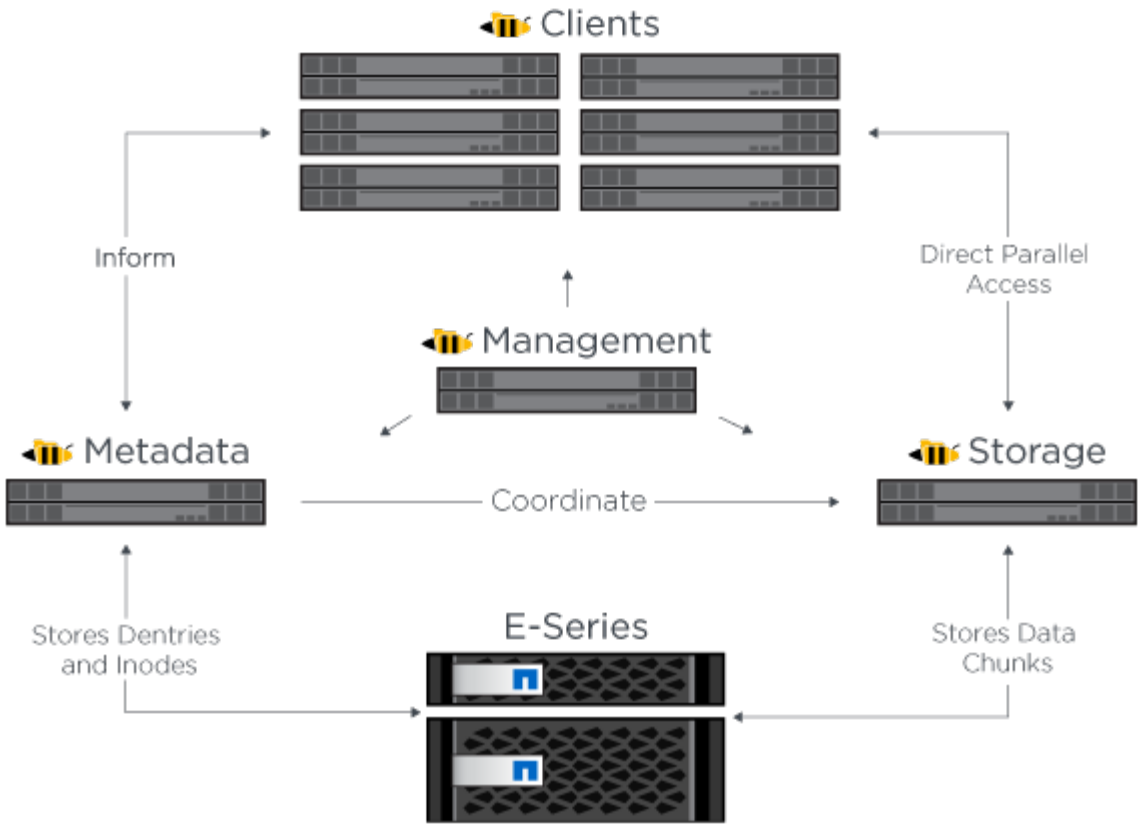
Para obtener información detallada, consulte Diseño de hardware verificado.

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.

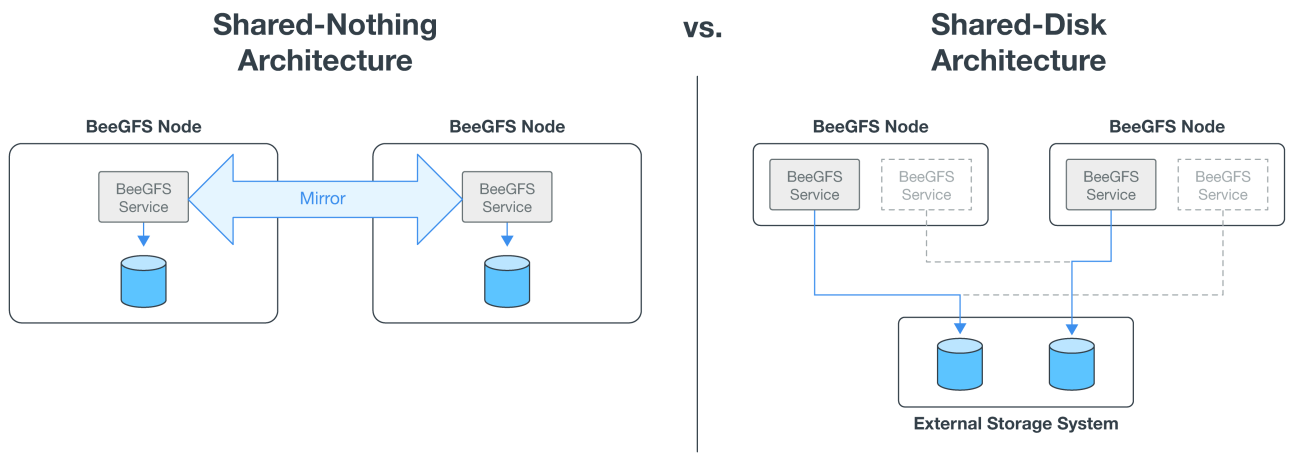
Los cuatro servicios principales se ejecutan en una amplia gama de distribuciones de Linux compatibles y se comunican a través de cualquier red compatible con TCP/IP o RDMA, incluidas InfiniBand (IB), Omni-Path (OPA) y RDMA over Converged Ethernet (roce). Los servicios de servidor BeeGFS (gestión, almacenamiento 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".

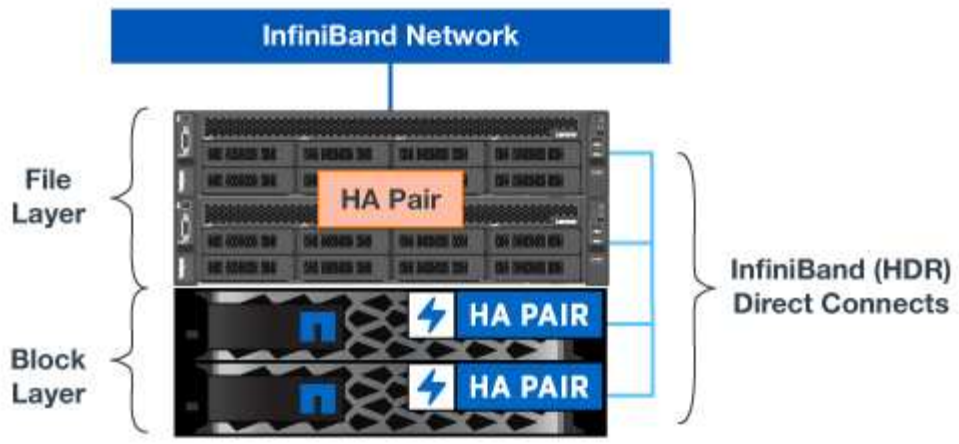
Nodos verificados

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

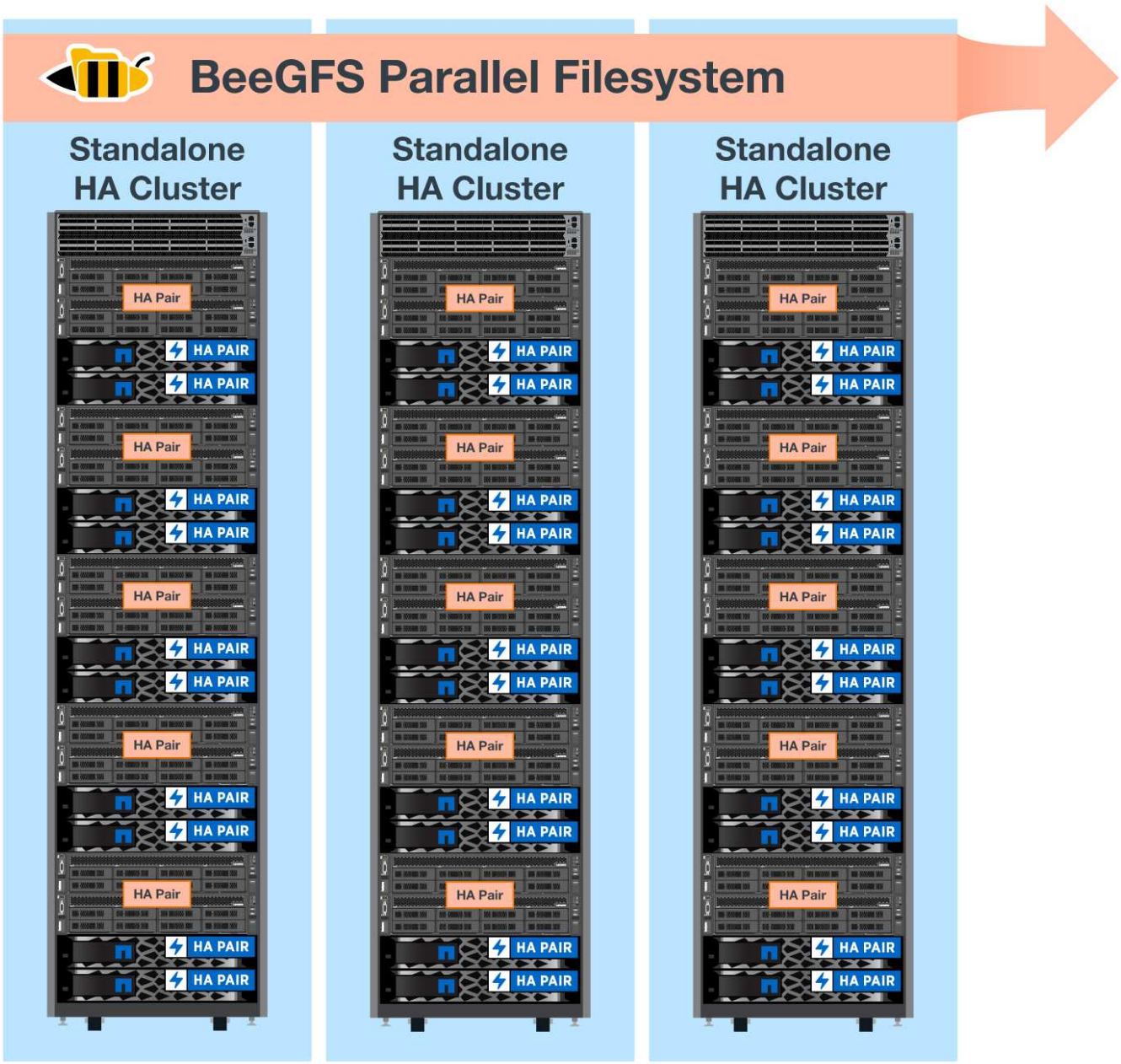
Los elementos básicos de la solución (que se muestran en la siguiente figura) utilizan los servidores de nodos de archivo verificados para la capa de archivo BeeGFS y dos sistemas de almacenamiento EF600 como capa de bloques.



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 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.



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	Cada nodo de archivo debe cumplir o superar las especificaciones de los nodos de archivo recomendados para lograr el rendimiento esperado. **Opciones de nodo de archivo recomendadas:** • **Lenovo ThinkSystem SR665 V3** ◦ **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:** ▪ 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 • **Lenovo ThinkSystem SR665** ◦ **Procesadores:** 2x AMD EPYC 7343 16C 3,2 GHz (configurado como dos zonas NUMA). ◦ **Memoria:** 256GB (16x 16GB TruDDR4 3200MHz RDIMM-A) ◦ **Expansión PCIe:** Cuatro ranuras PCIe Gen4 x16 (dos por zona NUMA) ◦ **Miscelánea:** ▪ 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.

Cantidad	Componente de hardware	Requisitos
8	Adaptadores de tarjeta de host InfiniBand (para nodos de archivo).	Los adaptadores de tarjeta de host variarán según el modelo de servidor utilizado para el nodo de archivo. Las recomendaciones para los nodos de archivo verificados incluyen: • **Servidor Lenovo ThinkSystem SR665 V3:** ◦ MCX755106AS-HEAT ConnectX-7, NDR200, QSFP112, 2 puertos, PCIe Gen5 x16, adaptador InfiniBand • **Servidor Lenovo ThinkSystem SR665:** ◦ MCX653106A-HDAT ConnectX-6, HDR, QSFP-56, 2 puertos, PCIe Gen4 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: • **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	Cantidad	Tipo de cable	Número de pieza
NVIDIA QM9700	4	Fibra activa	MFA7U10-H002 (cable de fibra activa NVIDIA, InfiniBand 400GB/s a 2x 200GB/s, OSFP a 2x QSFP56)
NVIDIA QM9700	4	Cobre pasivo	MCP7Y60-H002 (cable de cobre pasivo NVIDIA, InfiniBand de 400GB Gb/s a 2x 200GB Gb/s, OSFP a 2x QSFP56 m)
NVIDIA MQM8700	8	Fibra activa	MFS1S00-H003E (cable de fibra activa NVIDIA, InfiniBand 200GB Gb/s, QSFP56 m)
NVIDIA MQM8700	8	Cobre pasivo	MCP1650-H002E26 (cable de cobre pasivo NVIDIA, InfiniBand 200GB Gb/s, QSFP56 m)

Requisitos de software

Para obtener un rendimiento y una fiabilidad predecibles, los lanzamientos de BeeGFS en la solución de NetApp se prueban con versiones específicas de los componentes de software necesarios para implantar la solución.

Requisitos del nodo de archivo

De NetApp	Versión
Red Hat Enterprise Linux	Redhat 9.3 Server físico con alta disponibilidad (2 sockets). Los nodos de archivo requieren una suscripción válida a RedHat Enterprise Linux Server y el complemento de alta disponibilidad de Red Hat Enterprise Linux.
Kernel de Linux	5.14.0-362.24.1.el9_3.x86_64
Controladores InfiniBand/RDMA	MLNX_OFED_LINUX-23,10-3,2.2,0-LTS
Firmware de HCA	**ConnectX-7 HCA Firmware** FW: 28.39.1002 + PXE: 3.7.0201 + UEFI: 14.32.0012 **ConnectX-6 HCA Firmware** FW: 20.31.1014 + PXE: 3.6.0403 + UEFI: 14.24.0013

Requisitos del nodo de bloques de EF600

De NetApp	Versión
Sistema operativo SANtricity	11.80.0
NVSRAM	N6000-880834-D08.dlp
Firmware de la unidad	La última versión disponible para los modelos de unidad en uso.

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	7.4.4
Corosync	3.1.5-4
Marcapasos	2.1.4-5
OpenSM	Opensm-5.17.2 (de MLNX_OFED_LINUX-23,10-3,2.2,0-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	6.x cuando se instala mediante pip: Ansible-6.0.0 y ansible-core >= 2.13.0
Python	3,9 (o posterior)
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

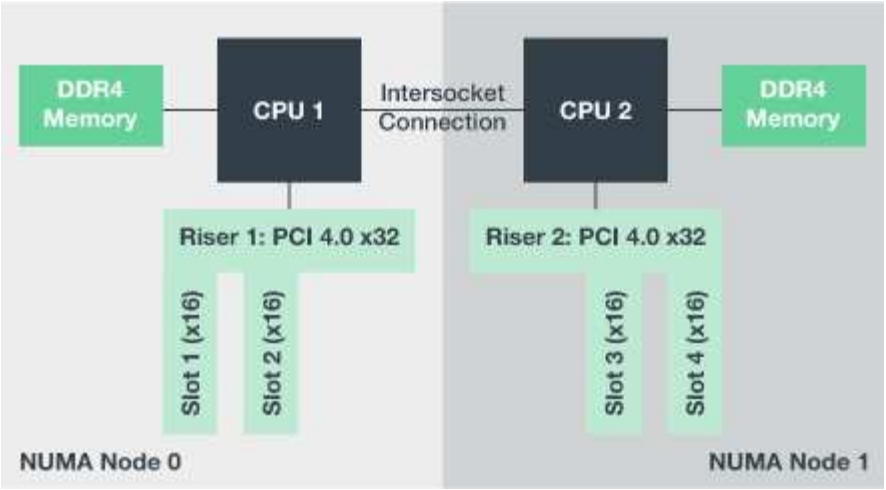
La configuración de hardware para BeeGFS en NetApp incluye nodos de archivo y cableado de red.

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.



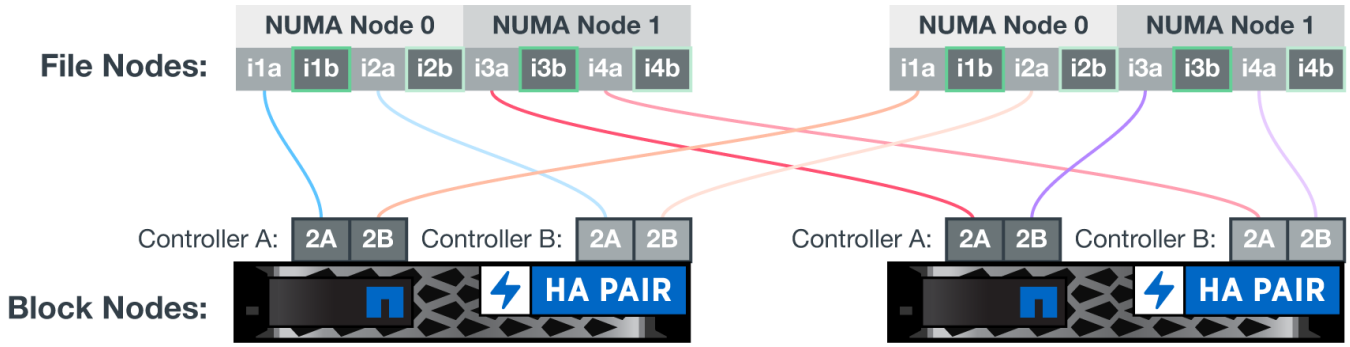
Los procesos BeeGFS se anclan a una zona NUMA en particular para garantizar que las interfaces utilizadas se encuentren en la misma zona. Esta configuración evita la necesidad de acceso remoto a través de la conexión entre sockets. La conexión entre zócalos se conoce a veces como el enlace QPI o GMI2; 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 divisores 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.

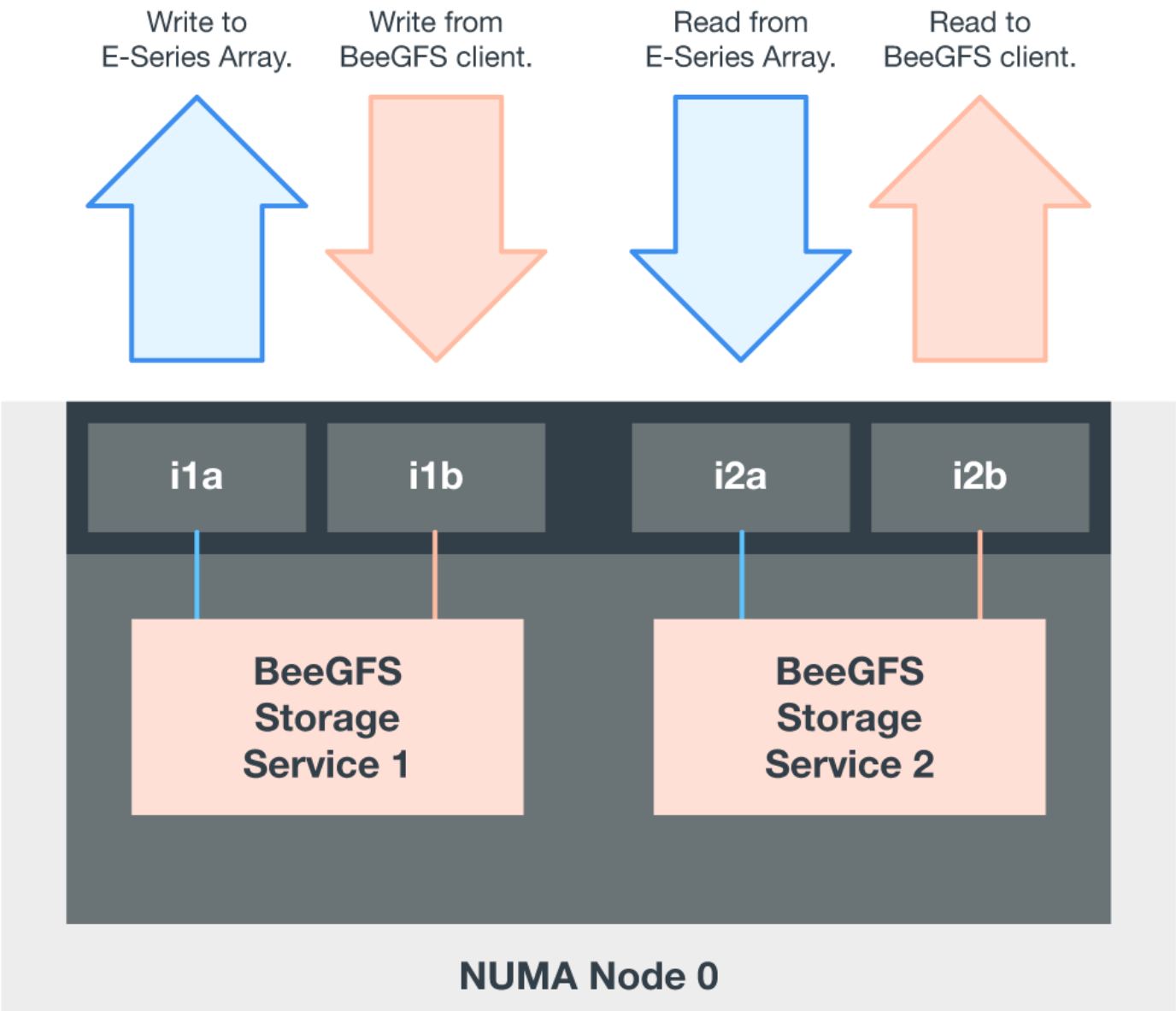
La configuración del cableado que se muestra en la figura permite a cada servicio BeeGFS:

- 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 conmute 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.

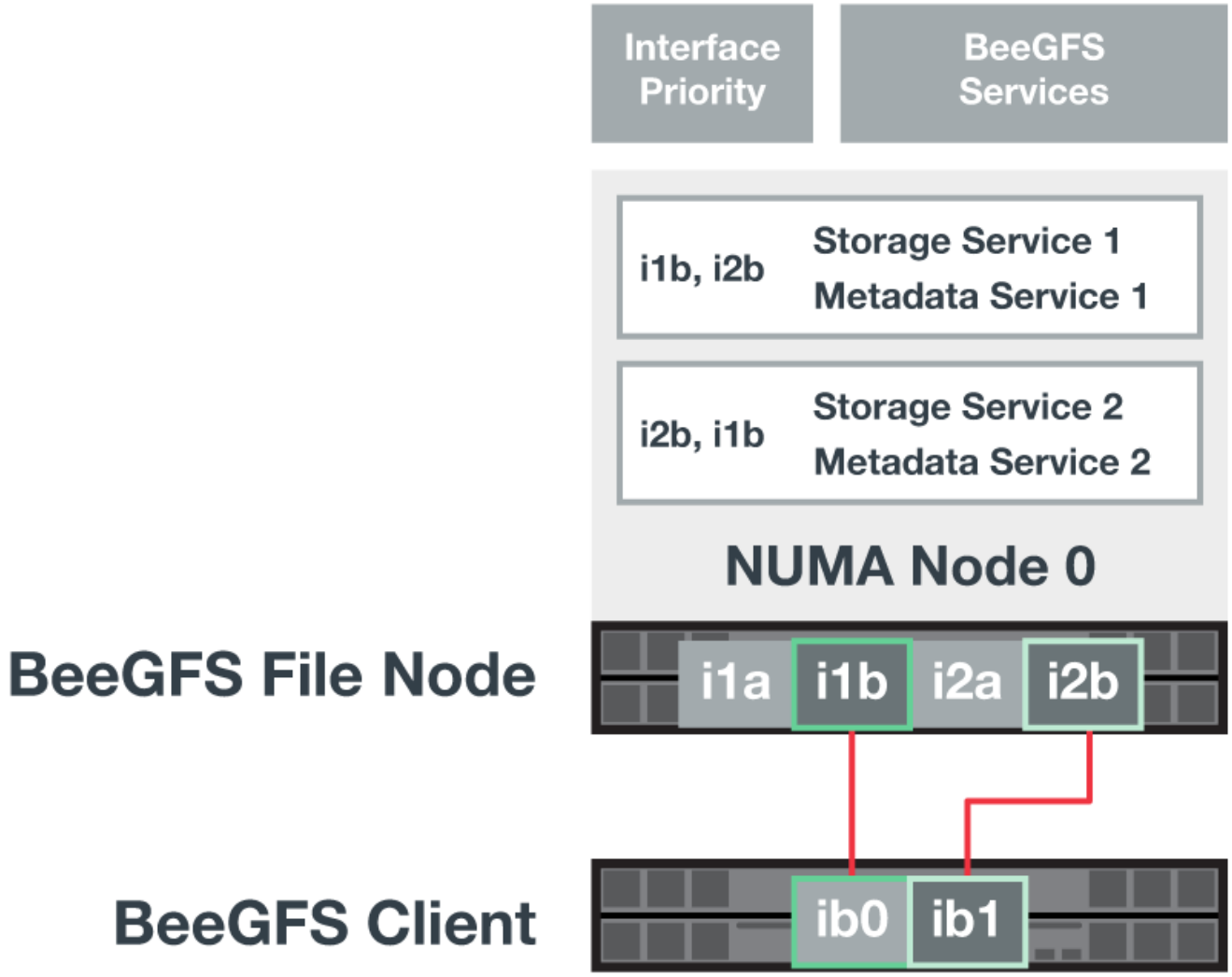
  La puesta en marcha basada en Ansible gestiona el apriete del comportamiento de la ruta inversa (RP) y del protocolo de resolución de direcciones (ARP), junto con la garantía de cuándo se inician y se detienen las IP flotantes; las reglas y rutas IP correspondientes se crean de forma dinámica para permitir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

+

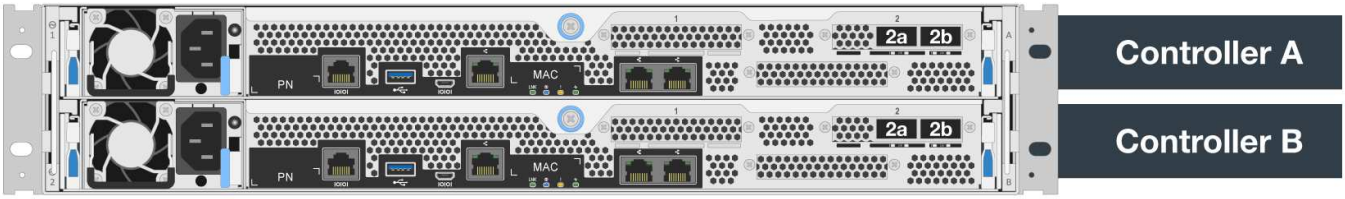
```
root@beegfs01:/mnt/beegfs# beegfs-ctl --getentryinfo myfile
Entry type: file
EntryID: 11D-624759A9-65
Metadata node: meta_01_tgt_0101 [ID: 101]
Stripe pattern details:
+ Type: RAID0
+ Chunksize: 1M
+ Number of storage targets: desired: 4; actual: 4
+ Storage targets:
  + 101 @ stor_01_tgt_0101 [ID: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

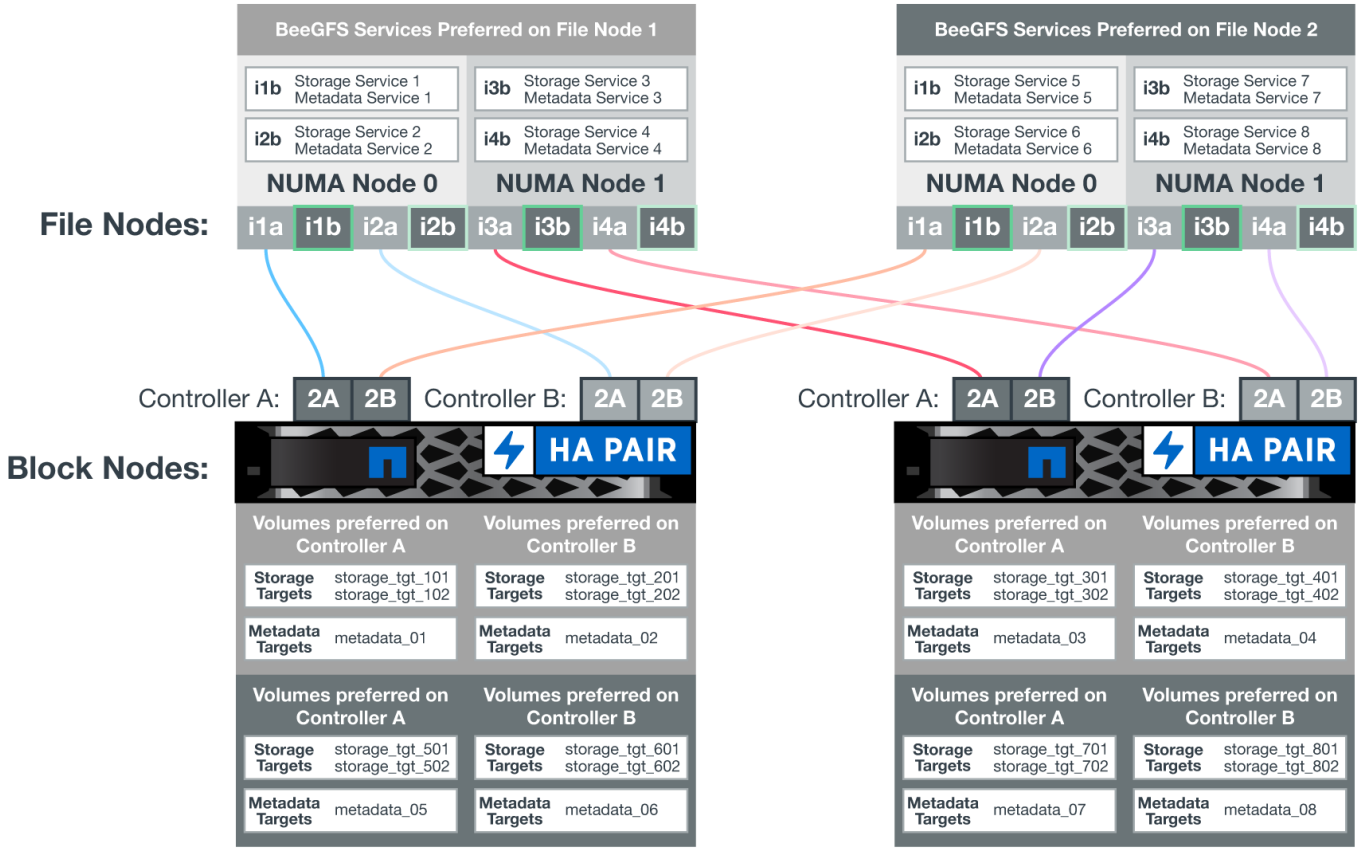
Los nodos de bloques constan de dos controladoras RAID activo/activo con acceso compartido al mismo conjunto de unidades. Por lo general, cada controladora tiene la mitad de los volúmenes configurados en el sistema, pero puede sustituir la otra controladora según sea necesario.

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 conmutación 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.

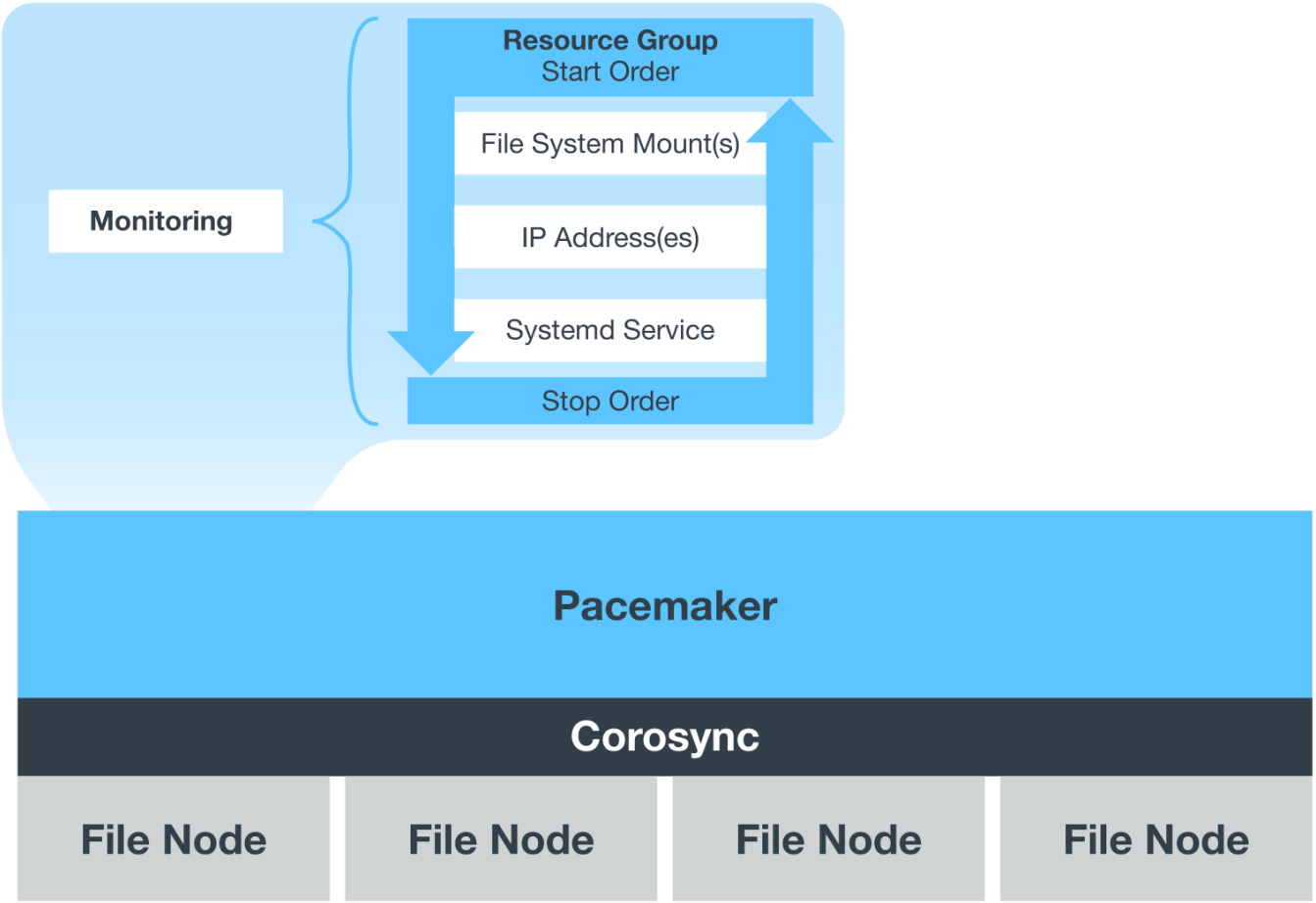
Si se produce un fallo en el clúster, Pacemaker reacciona y reinicia los recursos de BeeGFS en otro nodo. En algunos casos, es posible que Pacemaker no pueda comunicarse con el nodo defectuoso original para confirmar que los recursos están detenidos. Para verificar que el nodo está inactivo antes de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

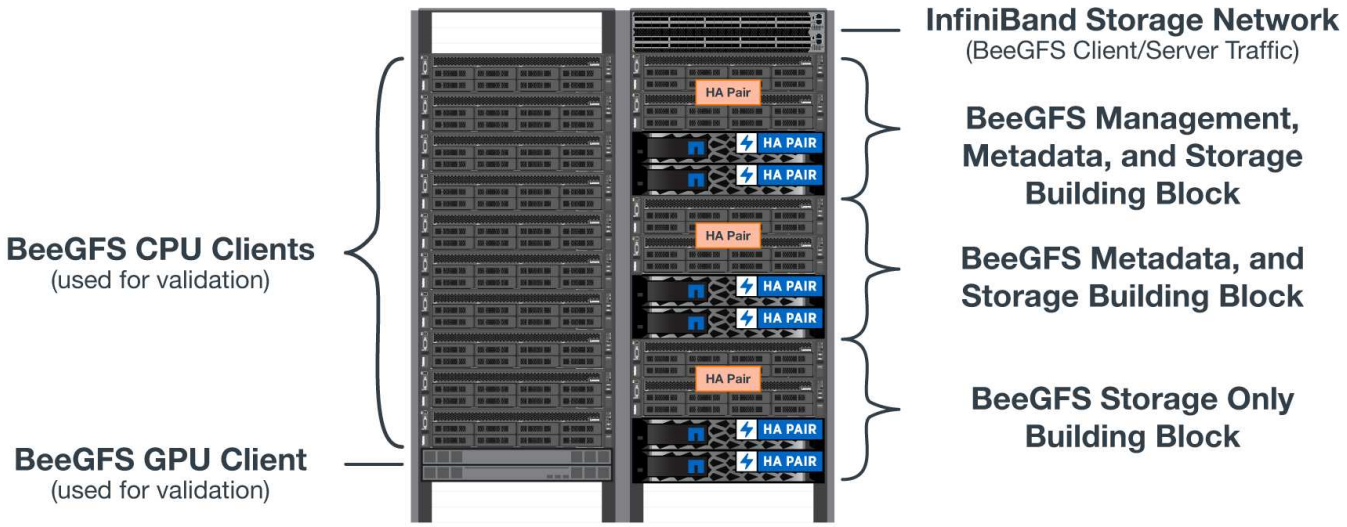
El diseño de segunda generación de BeeGFS en la solución de NetApp se verificó mediante tres perfiles de configuración de bloques básicos.

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
```

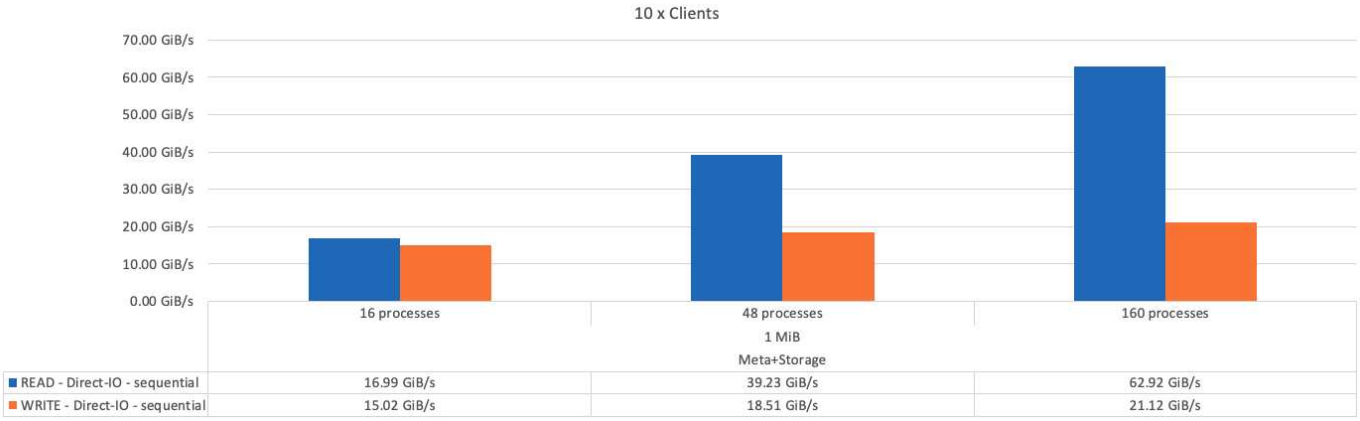
Un elemento básico de BeeGFS (gestión, metadatos y almacenamiento)

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.

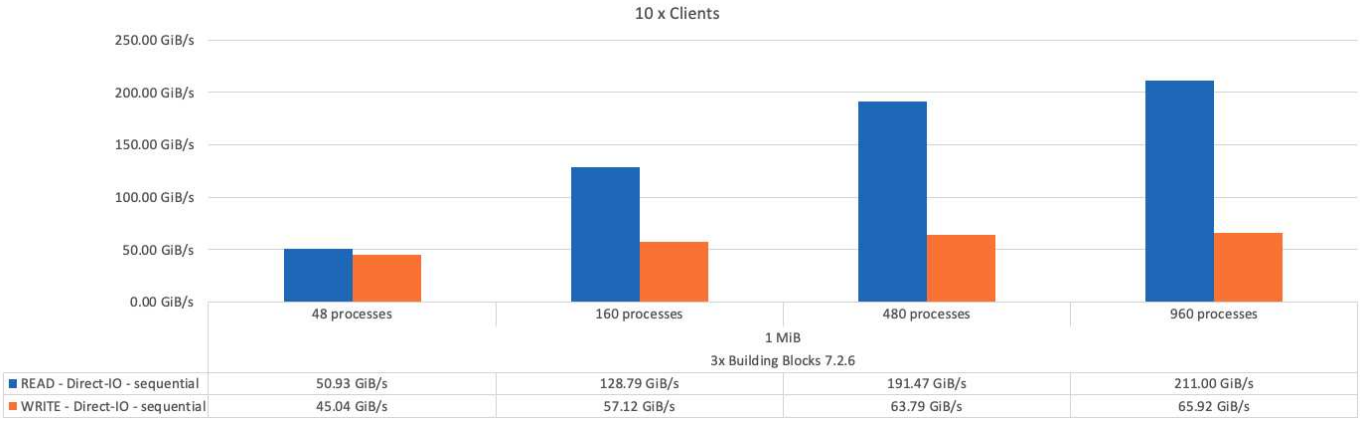


Elemento básico de sólo almacenamiento BeeGFS

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.

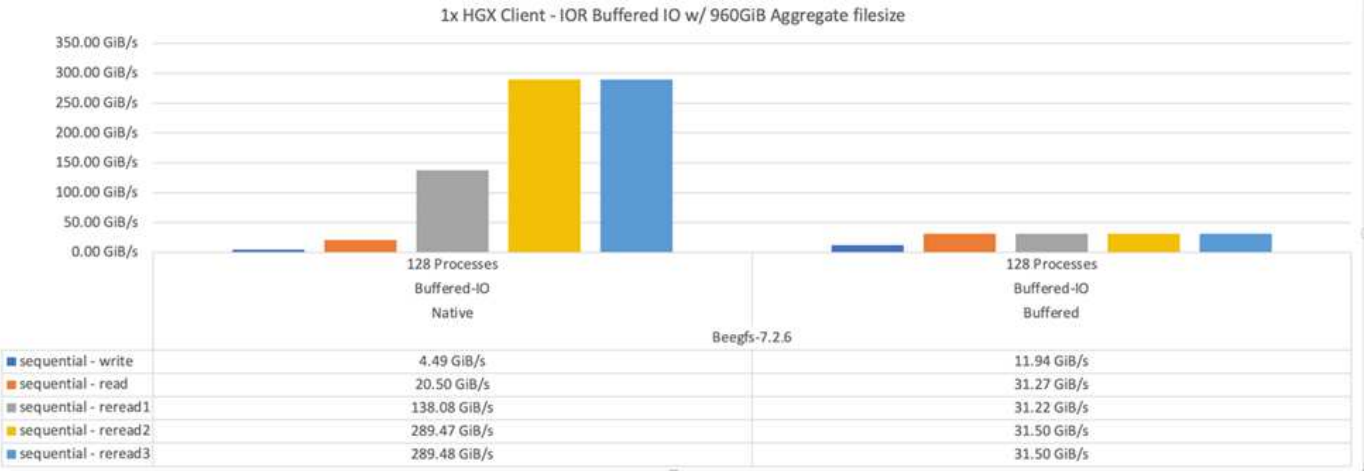
Pruebas de ancho de banda IOR: Un único cliente

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 relecan 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.

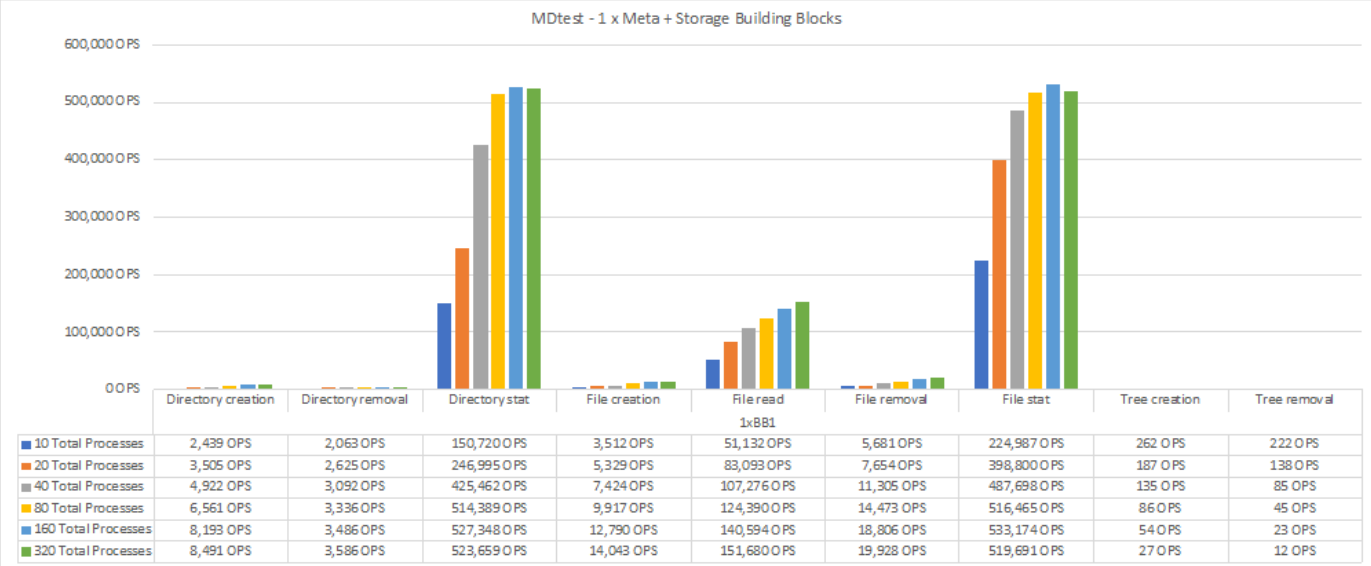
Se utilizaron los siguientes parámetros para ejecutar la prueba de referencia con el número total de procesos escalados de 10 a 320 en el paso del doble y con un tamaño de archivo de 4k.

```
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

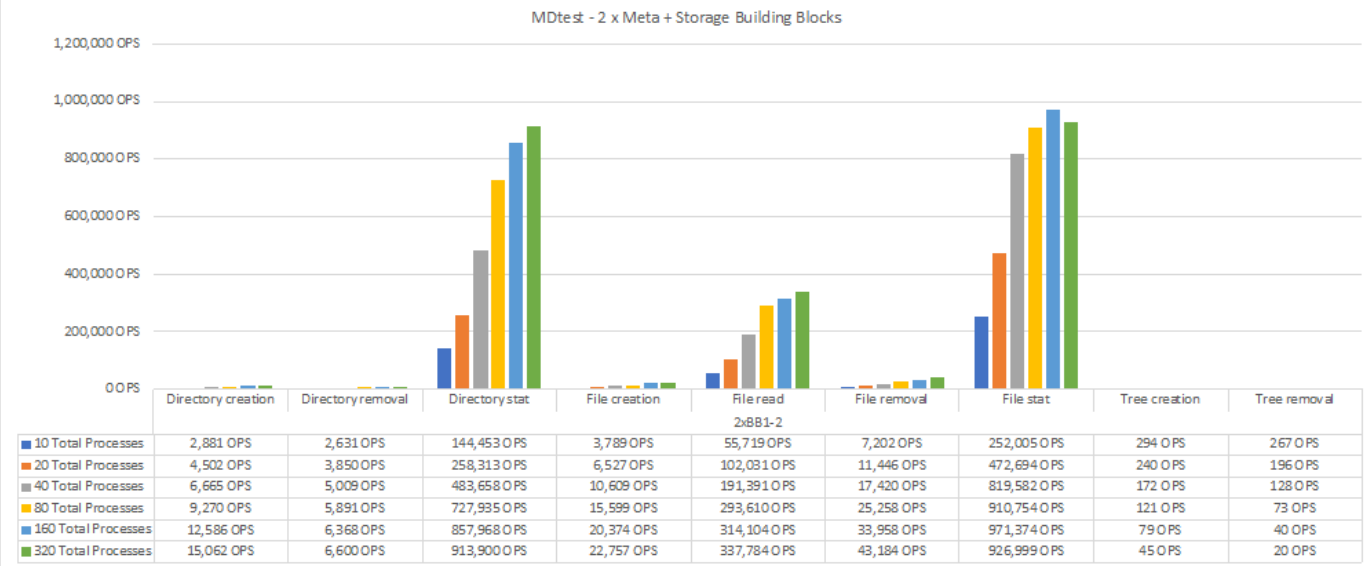
En el siguiente diagrama se muestran los resultados de MDTest con un bloque de creación de almacenamiento y metadatos BeeGFS.



1xBB1	Directory creation	Directory removal	Directory stat	File creation	File read	File removal	File stat	Tree creation	Tree removal
10 Total Processes	2,439 OPS	2,063 OPS	150,720 OPS	3,512 OPS	51,132 OPS	5,681 OPS	224,987 OPS	262 OPS	222 OPS
20 Total Processes	3,505 OPS	2,625 OPS	246,995 OPS	5,329 OPS	83,093 OPS	7,654 OPS	398,800 OPS	187 OPS	138 OPS
40 Total Processes	4,922 OPS	3,092 OPS	425,462 OPS	7,424 OPS	107,276 OPS	11,305 OPS	487,698 OPS	135 OPS	85 OPS
80 Total Processes	6,561 OPS	3,336 OPS	514,389 OPS	9,917 OPS	124,390 OPS	14,473 OPS	516,465 OPS	86 OPS	45 OPS
160 Total Processes	8,193 OPS	3,486 OPS	527,348 OPS	12,790 OPS	140,594 OPS	18,806 OPS	533,174 OPS	54 OPS	23 OPS
320 Total Processes	8,491 OPS	3,586 OPS	523,659 OPS	14,043 OPS	151,680 OPS	19,928 OPS	519,691 OPS	27 OPS	12 OPS

Dos metadatos BeeGFS + elementos básicos de almacenamiento

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



	Directory creation	Directory removal	Directory stat	File creation	File read	File removal	File stat	Tree creation	Tree removal
					2xBB1-2				
10 Total Processes	2,881 OPS	2,631 OPS	144,453 OPS	3,789 OPS	55,719 OPS	7,202 OPS	252,005 OPS	294 OPS	267 OPS
20 Total Processes	4,502 OPS	3,850 OPS	258,313 OPS	6,527 OPS	102,031 OPS	11,446 OPS	472,694 OPS	240 OPS	196 OPS
40 Total Processes	6,665 OPS	5,009 OPS	483,658 OPS	10,609 OPS	191,391 OPS	17,420 OPS	819,582 OPS	172 OPS	128 OPS
80 Total Processes	9,270 OPS	5,891 OPS	727,935 OPS	15,599 OPS	293,610 OPS	25,258 OPS	910,754 OPS	121 OPS	73 OPS
160 Total Processes	12,586 OPS	6,368 OPS	857,968 OPS	20,374 OPS	314,104 OPS	33,958 OPS	971,374 OPS	79 OPS	40 OPS
320 Total Processes	15,062 OPS	6,600 OPS	913,900 OPS	22,757 OPS	337,784 OPS	43,184 OPS	926,999 OPS	45 OPS	20 OPS

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 conmutación 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.

> 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:

1. **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 archivo Lenovo SR665 validados, consulte "Ajuste la configuración del sistema del nodo de archivos para aumentar el rendimiento".

2. **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

Puede poner en marcha BeeGFS en NetApp para nodos de archivos y bloques validados mediante la segunda generación del diseño de bloques básicos BeeGFS de NetApp.

Colecciones y roles de Ansible

BeeGFS se pone en marcha en una solución de NetApp con Ansible, un motor DE automatización TECNOLÓGICA más popular que se utiliza para automatizar la puesta en marcha de aplicaciones. Ansible utiliza una serie de archivos conocidos colectivamente como un inventario, que modela el sistema de archivos BeeGFS que desea implementar.

Ansible permite a empresas como NetApp ampliar la funcionalidad incorporada mediante colecciones en Ansible Galaxy (consulte "Colección BeeGFS de NetApp E-Series"). Las colecciones incluyen módulos que realizan alguna función o tarea específica (como crear un volumen E-Series) e incluyen roles que pueden llamar a varios módulos y otras funciones. 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 la incorporación de elementos básicos para ampliar los sistemas de archivos existentes.

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 metadatos y bloques básicos de almacenamiento o de elementos básicos solo de servicios de almacenamiento pueden variar en los procedimientos, según 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 especial atención a cualquier comentario en línea indicado por 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 la puesta en marcha, asegúrese de comprender cómo usar Ansible para configurar y poner en marcha BeeGFS en la solución de NetApp con el diseño de elementos básicos de segunda generación de BeeGFS.

El inventario de Ansible define la configuración para los nodos de archivos y bloques, y representa el sistema de archivos BeeGFS que desea poner en marcha. El inventario incluye hosts, grupos y variables que describen el sistema de archivos BeeGFS deseado. Los inventarios de muestras se pueden descargar desde "E-Series BeeGFS GitHub de NetApp".

Módulos y roles de Ansible

Para aplicar la configuración descrita en el inventario de Ansible, use los distintos módulos y roles de Ansible que se proporcionan en la colección de Ansible de NetApp E-Series, en particular el rol BeeGFS HA 7,4 (disponible en "E-Series BeeGFS GitHub de NetApp"), que pone 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

Use la estructura de inventario de Ansible para definir un clúster de alta disponibilidad de BeeGFS.

Cualquier persona con experiencia de Ansible anterior debe tener en cuenta que el rol de ha de BeeGFS implementa un método personalizado para descubrir qué variables (o hechos) se aplican a cada host. Esto es necesario para simplificar la creación de un inventario de Ansible que describa los recursos que se pueden ejecutar en varios servidores.

Un inventario de Ansible suele consistir en los archivos de `host_vars` y.. `group_vars`, y 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. Para obtener más información, consulte esta publicación de blog: "NetApp acelera la puesta en marcha de alta disponibilidad para BeeGFS con Ansible".

> 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.

Conexiones 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

Nodo de archivo	Puerto IB	Dirección IP	Nodo de bloques	Puerto IB	IP física	IP virtual
Impar (h1)	i2a	192.168.3.10	Impar (c1)	2 a	192.168.3.100	192.168.3.101
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 conmute 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).

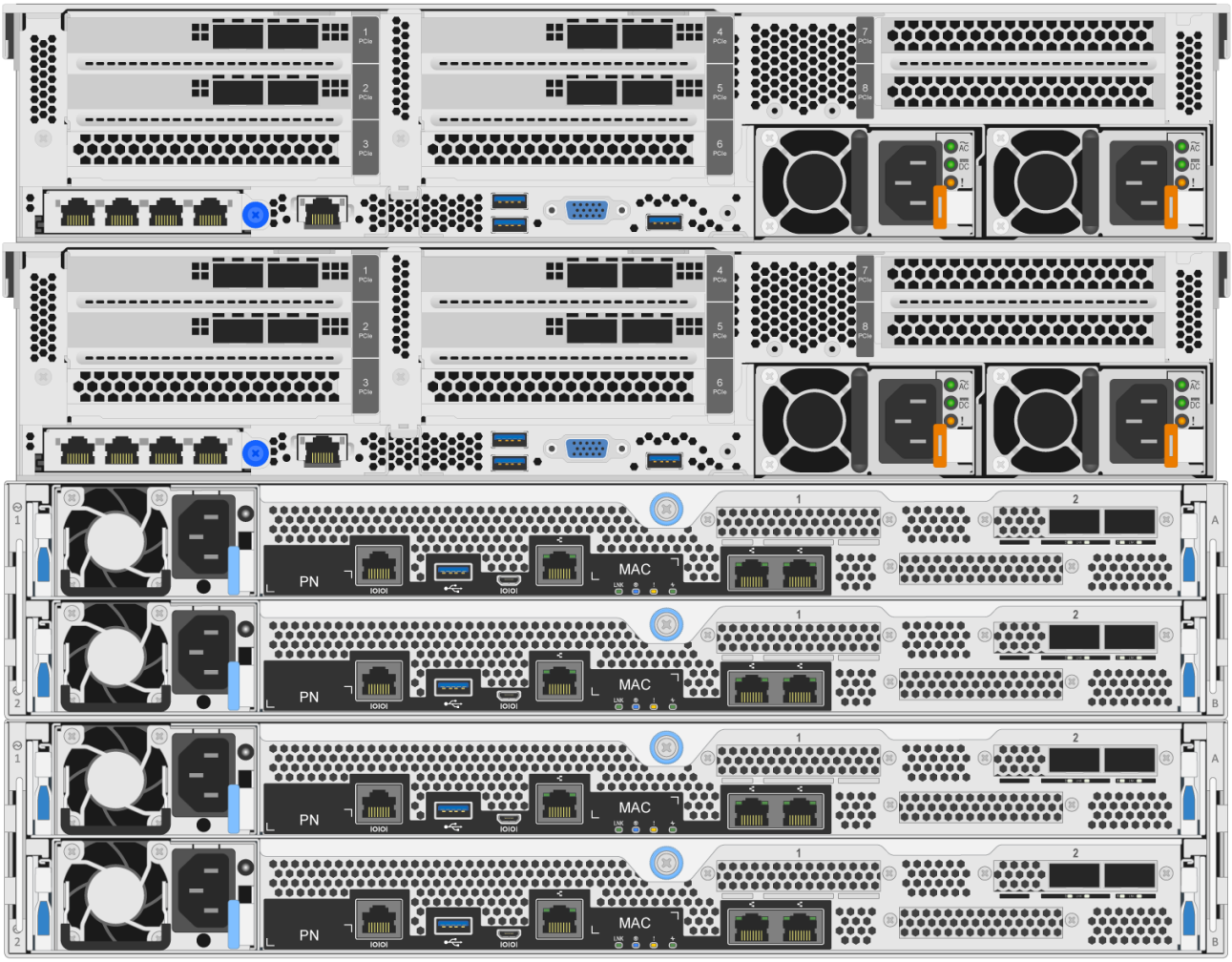
> Se necesita un mínimo de dos bloques de construcción para establecer el quórum en el clúster de conmutación por error. Un clúster de dos nodos tiene limitaciones que podrían impedir que se produzca una conmutación al respaldo correcta. Puede configurar un clúster de dos nodos incorporando un tercer 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

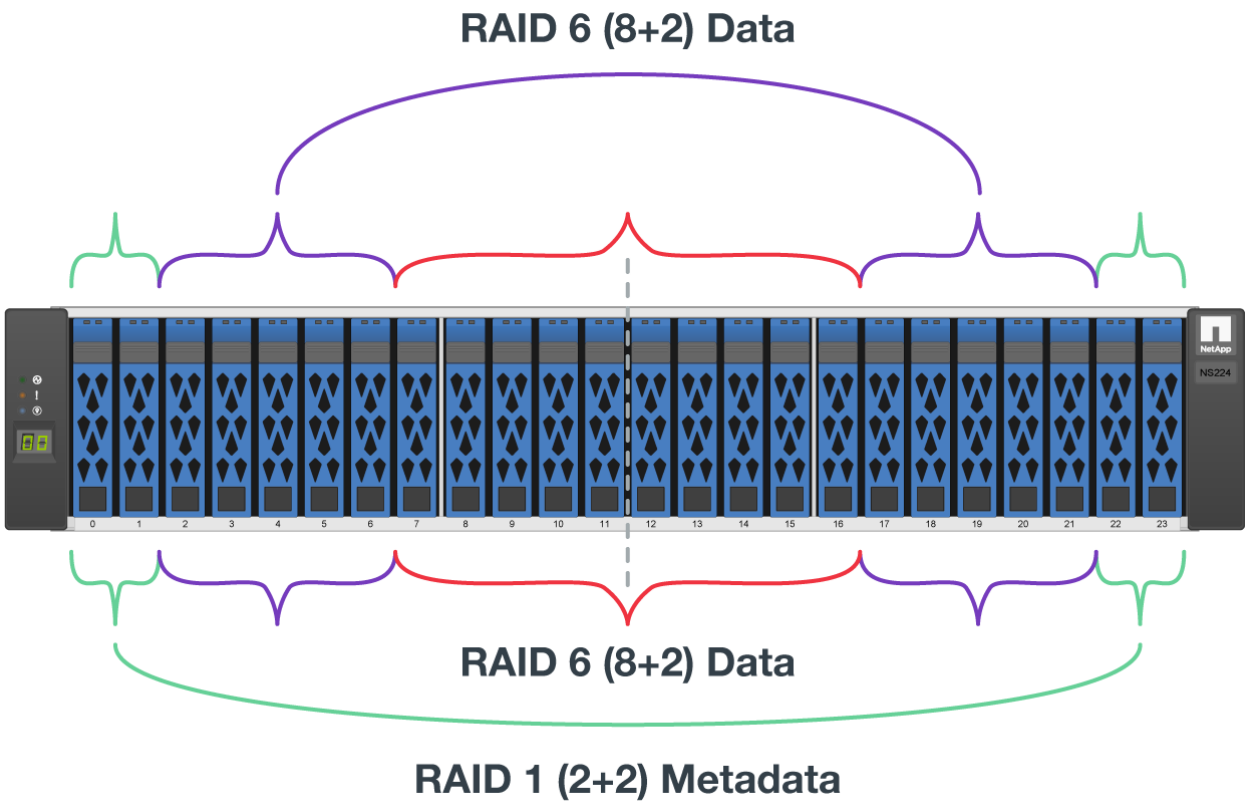
1. Configure cada nodo de archivo BeeGFS con cuatro adaptadores de canal de host (HCA) utilizando los modelos especificados en la "Requisitos técnicos". Inserte las HCA en las ranuras PCIe del nodo de archivo de acuerdo con las especificaciones siguientes:
   - **Servidor Lenovo ThinkSystem SR665 V3:** 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).

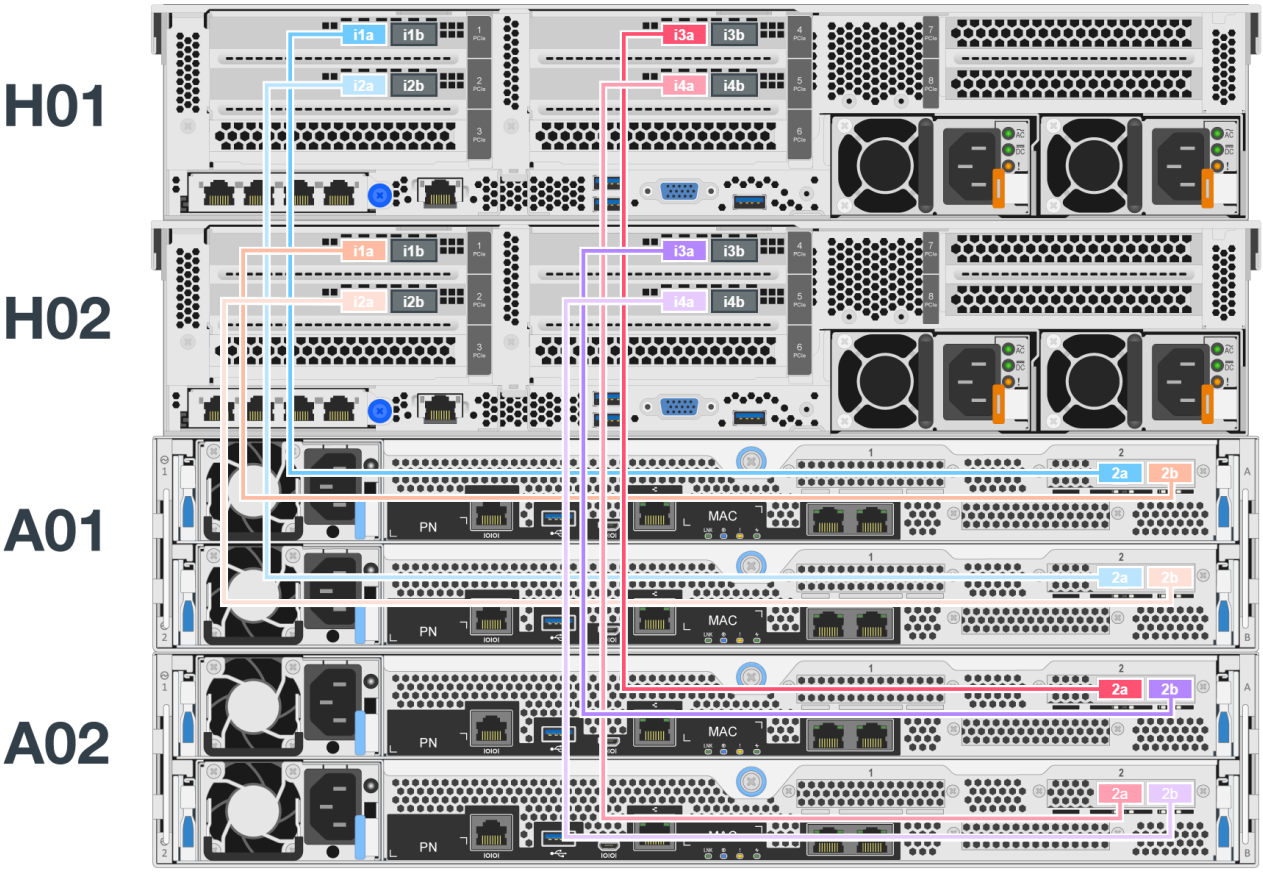


> La configuración de la fuente de alimentación para los casos de uso de producción normalmente debe utilizar fuentes de alimentación redundantes.

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.



4. 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.

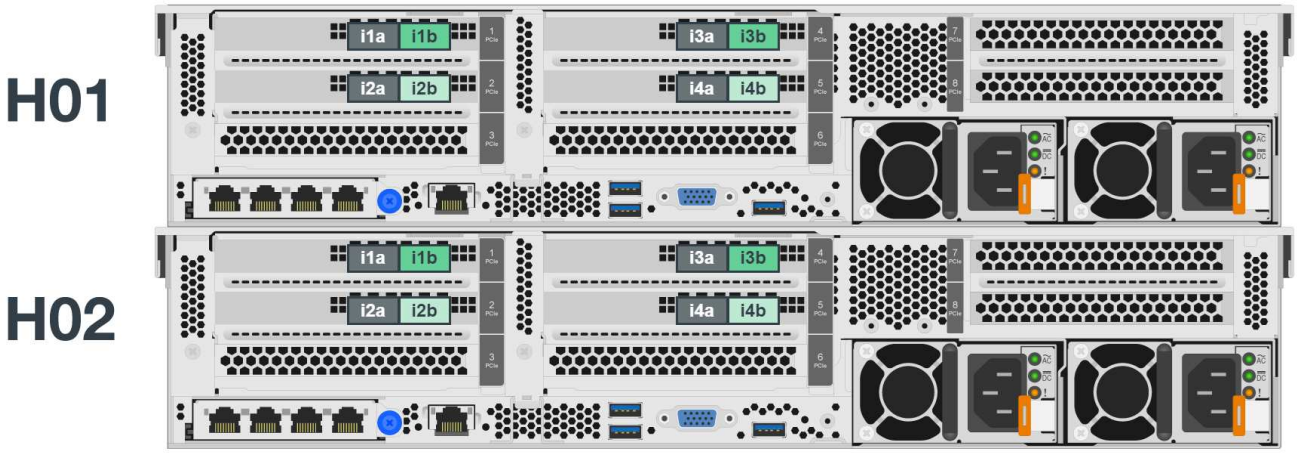


> Los nodos de varios elementos básicos nunca están conectados directamente. Cada bloque se debe tratar como una unidad independiente y toda la comunicación entre los bloques de construcción se produce a través de conmutadores de red.

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.



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 archivo Lenovo SR665 validados, consulte "Ajuste la configuración del sistema para obtener rendimiento".

3. Instale Red Hat 9.3 y configure el nombre de host y el puerto de red que se usan para gestionar 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 RedHat 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=9.3`. 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".

   Esta actualización se puede realizar descargando y ejecutando una versión de la herramienta mlxup que incluye el firmware recomendado. Puede descargar esta herramienta desde "Mlxup: Utilidad de actualización y consulta" ("guía del usuario").

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 cómo configurar los 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 presumen 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 contiene numerosos parámetros de ajuste que se pueden establecer 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

Ajuste del sistema	Cambiar a
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 identificar una máquina virtual o física con acceso de red a los puertos de gestión de todos los nodos de archivos y bloques que puedan usarse para configurar la solución.

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. Instale Python 3,10 y asegúrese de que está instalada la versión correcta de `pip` .

   ```
   sudo apt install python3.10 -y
   sudo apt install python3-pip
   sudo apt install sshpass
   ```

2. Crea enlaces simbólicos, asegurándote de que el binario Python 3,10 se use cuando `python3` o `python` sea llamado.

```
sudo ln -sf /usr/bin/python3.10 /usr/bin/python3
sudo ln -sf /usr/bin/python3 /usr/bin/python
```

3. Instale los paquetes Python necesarios para las colecciones de BeeGFS de NetApp.

```
python3 -m pip install ansible cryptography netaddr
```

> Para asegurarse de que está instalando una versión compatible de Ansible y todos los paquetes Python necesarios, consulte el archivo Léame de la colección BeeGFS. También se incluyen las versiones compatibles en "Requisitos técnicos".

4. Verifique que se hayan instalado las versiones correctas de Ansible y Python.

```
ansible --version
ansible [core 2.17.2]
  config file = None
  configured module search path = ['/root/.ansible/plugins/modules',
'/usr/share/ansible/plugins/modules']
  ansible python module location = /usr/local/lib/python3.10/dist-
packages/ansible
  ansible collection location =
/root/.ansible/collections:/usr/share/ansible/collections
  executable location = /usr/local/bin/ansible
  python version = 3.10.12 (main, Jul 29 2024, 16:56:48) [GCC 11.4.0]
(/usr/bin/python3)
  jinja version = 3.1.4
  libyaml = True
```

5. Almacene los inventarios de Ansible utilizados para describir la implementación de BeeGFS en sistemas de control de código fuente como Git o Bitbucket y, a continuación, instale Git para interactuar con esos sistemas.

```
sudo apt install git -y
```

6. Configure SSH sin contraseñas. Esta es la forma más sencilla de permitir que Ansible acceda a los nodos de archivos BeeGFS remotos desde el nodo de control de Ansible.
   a. En el nodo de control Ansible, si es necesario, genere un par de claves públicas con `ssh-keygen`
   b. Configure SSH sin contraseñas para cada uno de los nodos de archivos que utilizan `ssh-copy-id <ip_or_hostname>`

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

7. Utilice Ansible Galaxy para instalar la versión de la colección BeeGFS que se indica en "Requisitos

técnicos".

Esta instalación incluye dependencias adicionales de Ansible, como el software SANtricity de NetApp y las colecciones de hosts.

```
ansible-galaxy collection install netapp_eseries.beegfs:==3.2.0
```

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`

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 rellenar 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 desee 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 addressess 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: root
   ansible_become_password: <PASSWORD>
   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"
   ```

> Especialmente en entornos de producción, no almacene contraseñas en texto sin formato. En su lugar, utilice Ansible Vault (consulte "Cifrado de contenido con Ansible Vault") o el `--ask-become-pass` al ejecutar el libro de estrategia. Si la `ansible_ssh_user` ya lo es `root`, puede omitir opcionalmente la `ansible_become_password`.

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: hacluster              # BeeGFS HA cluster
username.
beegfs_ha_cluster_password: hapassword             # BeeGFS HA cluster
username's password.
beegfs_ha_cluster_password_sha512_salt: randomSalt # BeeGFS HA cluster
username's password salt.
beegfs_ha_mgmtd_floating_ip: 100.127.101.0         # BeeGFS management
service IP address.
# Email Alerts Configuration
beegfs_ha_enable_alerts: True
beegfs_ha_alert_email_list: ["email@example.com"]  # E-mail recipient
list for notifications when BeeGFS HA resources change or fail.  Often a
distribution list for the team responsible for managing the cluster.
beegfs_ha_alert_conf_ha_group_options:
      mydomain: “example.com”
# The mydomain parameter specifies the local internet domain name. This
is optional when the cluster nodes have fully qualified hostnames (i.e.
host.example.com).
# Adjusting the following parameters is optional:
beegfs_ha_alert_timestamp_format: "%Y-%m-%d %H:%M:%S.%N" #%H:%M:%S.%N
beegfs_ha_alert_verbosity: 3
#  1) high-level node activity
#  3) high-level node activity + fencing action information + resources
(filter on X-monitor)
#  5) high-level node activity + fencing action information + resources
```

> 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.

3. 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, conmutan 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: <PDU_USERNAME>
     passwd: <PDU_PASSWORD>
     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>
  password: <BMC_PASSWORD>
  ssl_insecure: 1 # If a valid SSL certificate is not available
specify "1".
beegfs_ha_fencing_agents:
  fence_redfish:
    - pcmk_host_list: <HOSTNAME>
      ip: <BMC_IP>
      <<: *redfish
    - pcmk_host_list: <HOSTNAME>
      ip: <BMC_IP>
      <<: *redfish
# For details on configuring other fencing agents see
https://access.redhat.com/documentation/en-
us/red_hat_enterprise_linux/9/html/configuring_and_managing_high_avai
lability_clusters/assembly_configuring-fencing-configuring-and-
managing-high-availability-clusters.
```

4. 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: <PASSWORD>
eseries_validate_certs: false
```

> No se recomienda enumerar las contraseñas en texto sin formato. Use el almacén de Ansible o proporcione el `eseries_system_password` Cuando ejecute Ansible con `--extra-vars`.

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_nvsram: "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, rellenar 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:100.127.102.1/16	0	netapp_01	beegfs_m1_m2_m5_m6	A.
meta_02.yml	8025	i2b: 100.127.102.2/16 i1b:100.127.101.2/16	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:100.127.102.3/16	1	netapp_02	beegfs_m3_m4_m7_m8	A.
meta_04.yml	8045	i4b: 100.127.102.4/16 i3b:100.127.101.4/16	1	netapp_02	beegfs_m3_m4_m7_m8	B
meta_05.yml	8055	i1b: 100.127.101.5/16 i2b:100.127.102.5/16	0	netapp_01	beegfs_m1_m2_m5_m6	A.
meta_06.yml	8065	i2b: 100.127.102.6/16 i1b:100.127.101.6/16	0	netapp_01	beegfs_m1_m2_m5_m6	B
meta_07.yml	8075	i3b: 100.127.101.7/16 i4b:100.127.102.7/16	1	netapp_02	beegfs_m3_m4_m7_m8	A.
meta_08.yml	8085	i4b: 100.127.102.8/16 i3b:100.127.101.8/16	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:100.127.104.1/16	0	netapp_01	beegfs_s1_s2	A.
stor_02.yml	8023	i2b: 100.127.104.2/16 i1b:100.127.103.2/16	0	netapp_01	beegfs_s1_s2	B
stor_03.yml	8033	i3b: 100.127.103.3/16 i4b:100.127.104.3/16	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:100.127.103.4/16	1	netapp_02	beegfs_s3_s4	B
stor_05.yml	8053	i1b: 100.127.103.5/16 i2b:100.127.104.5/16	0	netapp_01	beegfs_s5_s6	A.
stor_06.yml	8063	i2b: 100.127.104.6/16 i1b:100.127.103.6/16	0	netapp_01	beegfs_s5_s6	B
stor_07.yml	8073	i3b: 100.127.103.7/16 i4b:100.127.104.7/16	1	netapp_02	beegfs_s7_s8	A.
stor_08.yml	8083	i4b: 100.127.104.8/16 i3b:100.127.103.8/16	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:100.127.102.9/16	0	netapp_03	beegfs_m9_m10_m13_m14	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:100.127.101.10/16	0	netapp_03	beegfs_m9_m10_m13_m14	B
meta_11.yml	8035	i3b: 100.127.101.11/16 i4b:100.127.102.11/16	1	netapp_04	beegfs_m11_m12_m15_m16	A.
meta_12.yml	8045	i4b: 100.127.102.12/16 i3b:100.127.101.12/16	1	netapp_04	beegfs_m11_m12_m15_m16	B
meta_13.yml	8055	i1b: 100.127.101.13/16 i2b:100.127.102.13/16	0	netapp_03	beegfs_m9_m10_m13_m14	A.
meta_14.yml	8065	i2b: 100.127.102.14/16 i1b:100.127.101.14/16	0	netapp_03	beegfs_m9_m10_m13_m14	B
meta_15.yml	8075	i3b: 100.127.101.15/16 i4b:100.127.102.15/16	1	netapp_04	beegfs_m11_m12_m15_m16	A.
meta_16.yml	8085	i4b: 100.127.102.16/16 i3b:100.127.101.16/16	1	netapp_04	beegfs_m11_m12_m15_m16	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 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_09.yml	8013	i1b: 100.127.103.9/16 i2b:100.127.104.9/16	0	netapp_03	beegfs_s9_s10	A.
stor_10.yml	8023	i2b: 100.127.104.10/16 i1b:100.127.103.10/16	0	netapp_03	beegfs_s9_s10	B
stor_11.yml	8033	i3b: 100.127.103.11/16 i4b:100.127.104.11/16	1	netapp_04	beegfs_s11_s12	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:100.127.103.12/16	1	netapp_04	beegfs_s11_s12	B
stor_13.yml	8053	i1b: 100.127.103.13/16 i2b:100.127.104.13/16	0	netapp_03	beegfs_s13_s14	A.
stor_14.yml	8063	i2b: 100.127.104.14/16 i1b:100.127.103.14/16	0	netapp_03	beegfs_s13_s14	B
stor_15.yml	8073	i3b: 100.127.103.15/16 i4b:100.127.104.15/16	1	netapp_04	beegfs_s15_s16	A.
stor_16.yml	8083	i4b: 100.127.104.16/16 i3b:100.127.103.16/16	1	netapp_04	beegfs_s15_s16	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. Pulg `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 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_17.yml	8013	i1b: 100.127.103.17/16 i2b:100.127.104.17/16	0	netapp_05	beegfs_s17_s18	A.
stor_18.yml	8023	i2b: 100.127.104.18/16 i1b:100.127.103.18/16	0	netapp_05	beegfs_s17_s18	B
stor_19.yml	8033	i3b: 100.127.103.19/16 i4b:100.127.104.19/16	1	netapp_06	beegfs_s19_s20	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:100.127.103.20/16	1	netapp_06	beegfs_s19_s20	B
stor_21.yml	8053	i1b: 100.127.103.21/16 i2b:100.127.104.21/16	0	netapp_05	beegfs_s21_s22	A.
stor_22.yml	8063	i2b: 100.127.104.22/16 i1b:100.127.103.22/16	0	netapp_05	beegfs_s21_s22	B
stor_23.yml	8073	i3b: 100.127.103.23/16 i4b:100.127.104.23/16	1	netapp_06	beegfs_s23_s24	A.
stor_24.yml	8083	i4b: 100.127.104.24/16 i3b:100.127.103.24/16	1	netapp_06	beegfs_s23_s24	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 en 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_mgmtd_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.

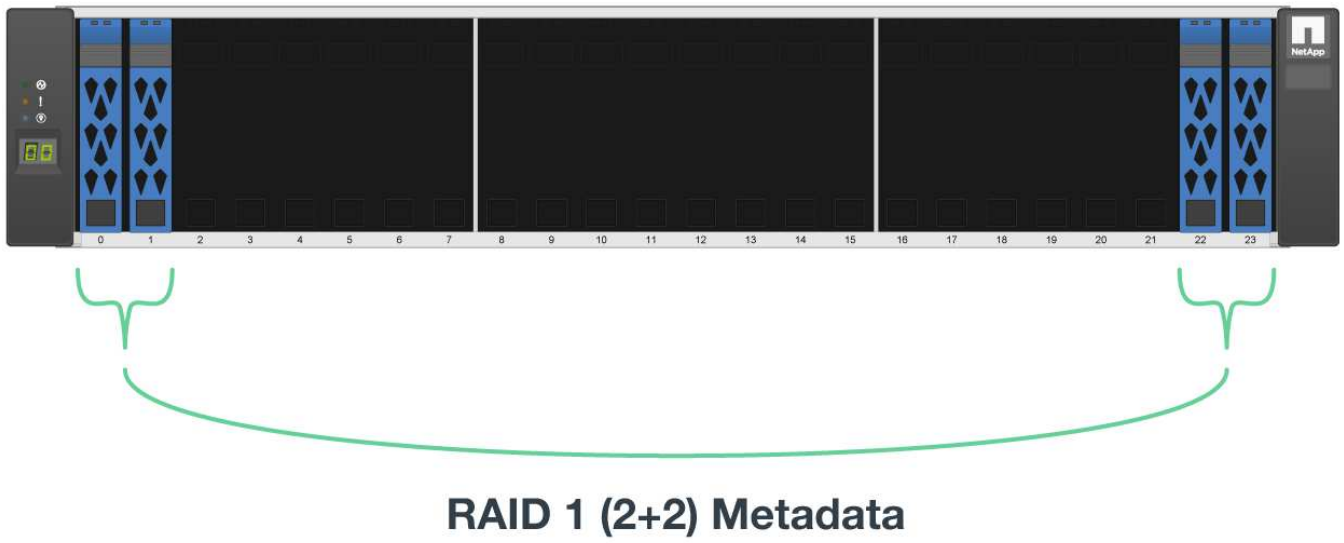
Controladoras

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".

Utilizar arquitecturas personalizadas

Descripción general y requisitos

Use cualquier sistema de almacenamiento E/EF-Series de NetApp como nodos de bloque BeeGFS y servidores x86 como nodos de archivos BeeGFS cuando ponga en marcha clústeres de alta disponibilidad BeeGFS con Ansible.

Las definiciones de terminología utilizadas en esta sección se pueden encontrar en la "términos y conceptos" página.

Introducción

Aunque"Arquitecturas verificadas de NetApp", proporcionan configuraciones de referencia predefinidas y directrices sobre dimensiones, algunos clientes y partners pueden preferir diseñar arquitecturas personalizadas que se adapten mejor a requisitos específicos o a las preferencias de hardware. Una de las principales ventajas de elegir BeeGFS en NetApp es la capacidad de poner en marcha clústeres de alta disponibilidad de disco compartido BeeGFS mediante Ansible, lo que simplifica la gestión del clúster y mejora la fiabilidad con componentes de alta disponibilidad creados por NetApp. La puesta en marcha de arquitecturas BeeGFS personalizadas en NetApp todavía se realiza con Ansible, lo que mantiene un enfoque similar al de los dispositivos en una gama flexible de hardware.

En esta sección, se describen los pasos generales necesarios para poner en marcha sistemas de archivos BeeGFS en hardware de NetApp y el uso de Ansible para configurar los sistemas de archivos BeeGFS. Para obtener información detallada sobre las mejores prácticas relacionadas con el diseño de sistemas de archivos BeeGFS y ejemplos optimizados, consulte la "Arquitecturas verificadas de NetApp"sección.

Visión General de la implementación

Generalmente, la implementación de un sistema de archivos BeeGFS incluye los siguientes pasos:

- Configuración inicial:
  - Instalar/cablear la tornillería.
  - Configure los nodos de archivos y bloques.
  - Configure un nodo de control Ansible.
- Defina el sistema de archivos BeeGFS como un inventario de Ansible.
- Ejecute Ansible contra los nodos de archivos y bloques para poner en marcha BeeGFS.
  - Opcionalmente para configurar clientes y montar BeeGFS.

Las siguientes secciones tratarán estos pasos con más detalle.

Ansible gestiona todas las tareas de configuración y aprovisionamiento del software, incluidas:

- Crear/asignar volúmenes en nodos de bloques.
- Formateo/ajuste de volúmenes en nodos de archivo.
- Instalar/configurar software en nodos de archivos.
- Establecer el clúster de alta disponibilidad y configurar los recursos de BeeGFS y los servicios del sistema de archivos.

Requisitos

Ya está disponible el soporte para BeeGFS en Ansible "Galaxia de ansible" Como un conjunto de funciones y módulos que automatizan la implementación y gestión integral de clústeres de alta disponibilidad de BeeGFS.

BeeGFS se versión siguiendo un esquema de control de versiones <major>.<minor>.<patch> y la colección mantiene roles para cada versión compatible de <major>.<minor> de BeeGFS, por ejemplo BeeGFS 7.2 o BeeGFS 7.3. A medida que se publican actualizaciones de la colección, la versión de revisión de cada rol se actualizará para señalar la última versión disponible de BeeGFS para esa rama de versión (por ejemplo: 7.2.8). Cada versión de la colección también está probada y compatible con distribuciones y versiones específicas de Linux, actualmente Red Hat para nodos de archivos, y RedHat y Ubuntu para clientes. No se admite la ejecución de otras distribuciones; no se recomienda ejecutar otras versiones (especialmente otras versiones principales).

Nodo de control de Ansible

Este nodo contendrá el inventario y los libros de estrategia utilizados para gestionar BeeGFS. Requiere:

- Ansible 6.x (núcleo de Ansible 2.13)
- Python 3.6 (o posterior)
- Paquetes de Python (pip): Ipaddr y netaddr

También es recomendable configurar SSH sin contraseñas desde el nodo de control en todos los clientes y nodos de archivos BeeGFS.

Nodos de archivo BeeGFS

Los nodos de archivo deben ejecutar RedHat 9.3 y tener acceso al repositorio ha que contenga los paquetes requeridos (marcapaso, corosync, valla-agents-all, Resource-agents). Por ejemplo, se puede ejecutar el siguiente comando para habilitar el repositorio apropiado en RedHat 9:

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

Nodos de cliente BeeGFS

Hay disponible un rol de Ansible para el cliente BeeGFS para instalar el paquete de cliente BeeGFS y gestionar los montajes BeeGFS. Esta función se ha probado con RedHat 8.4 y Ubuntu 22.04.

Si no utiliza Ansible para configurar el cliente BeeGFS y montar BeeGFS, any "Distribución y kernel de Linux compatibles con BeeGFS" puede utilizarse.

Configuración inicial

Instale y conecte el cableado de la tornillería

Pasos necesarios para instalar y cablear el hardware utilizado para ejecutar BeeGFS en NetApp.

Planifique la instalación

Cada sistema de archivos BeeGFS consistirá en un número determinado de nodos de archivo en los que se ejecutan servicios BeeGFS mediante el almacenamiento de fondo proporcionado por un número determinado de nodos de bloque. Los nodos de archivos están configurados en uno o varios clústeres de alta disponibilidad para proporcionar tolerancia a fallos en los servicios BeeGFS. Cada nodo de bloque ya es un par de alta disponibilidad activo-activo. El número mínimo de nodos de archivo admitidos en cada clúster de alta disponibilidad es de tres y el número máximo de nodos de archivo admitidos en cada clúster es de diez. Los sistemas de archivos BeeGFS pueden escalar más allá de diez nodos mediante la puesta en marcha de varios clústeres de alta disponibilidad independientes que funcionan en conjunto para proporcionar un espacio de nombres único del sistema de archivos.

Normalmente, cada clúster de alta disponibilidad se implementa como una serie de «elementos básicos» donde hay algún número de nodos de archivo (servidores x86) conectados directamente a un cierto número de nodos de bloques (normalmente sistemas de almacenamiento E-Series). Esta configuración crea un clúster asimétrico, en el que los servicios BeeGFS sólo pueden ejecutarse en determinados nodos de archivo que tienen acceso al almacenamiento de bloques de fondo utilizado para los destinos BeeGFS. El equilibrio de los nodos de archivo a bloque en cada elemento básico y el protocolo de almacenamiento que se utiliza para las conexiones directas dependen de los requisitos de una instalación concreta.

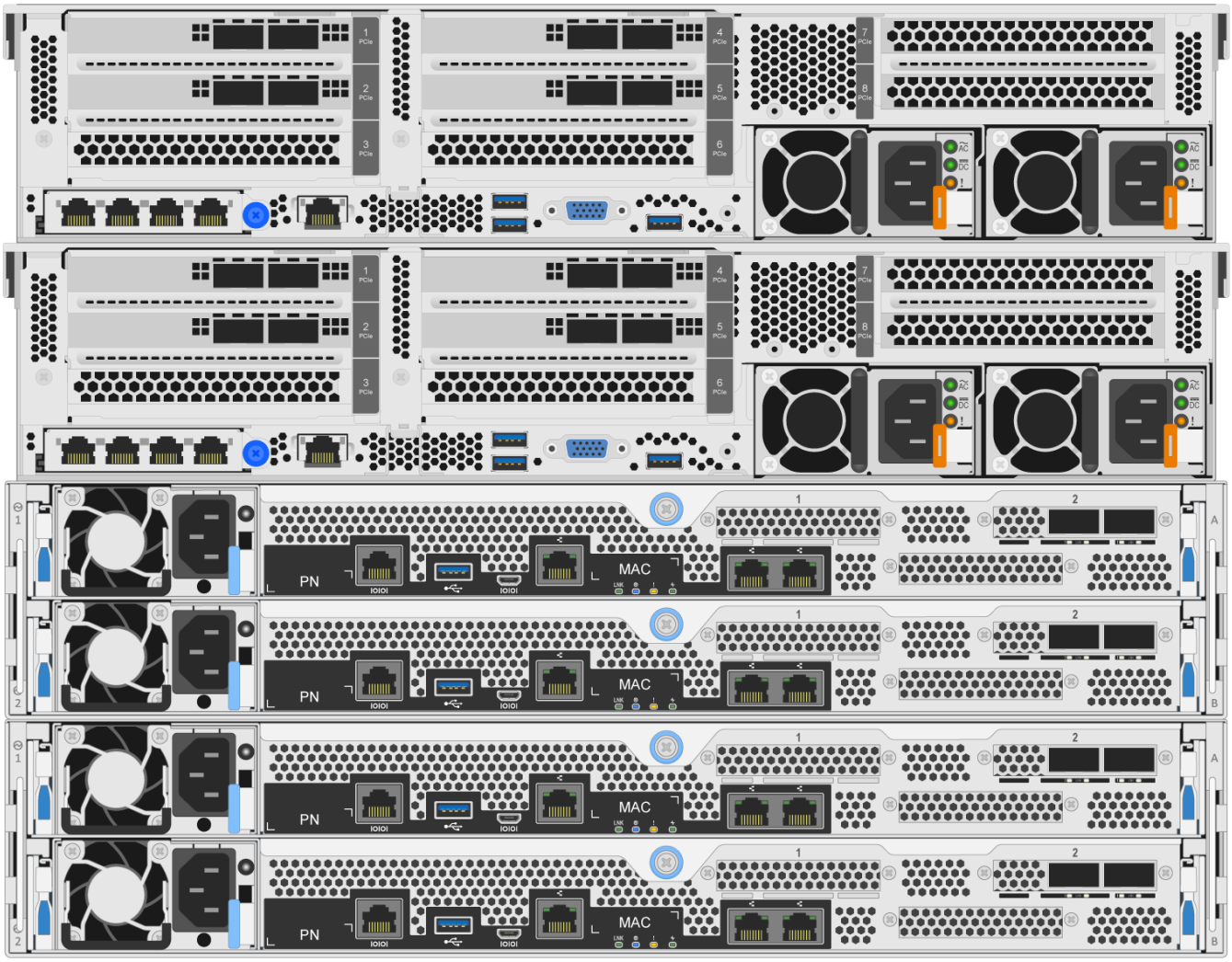
Una arquitectura de cluster de alta disponibilidad alternativa utiliza una estructura de almacenamiento (también conocida como red de área de almacenamiento o SAN) entre los nodos de archivo y de bloque para establecer un cluster simétrico. Esto permite que los servicios de BeeGFS se ejecuten en cualquier nodo de archivo en un clúster de alta disponibilidad en particular. Como los clústeres simétricos en general no son tan rentables debido al hardware SAN adicional, esta documentación presupone el uso de un clúster asimétrico desplegado como una serie de uno o más bloques de construcción.

> Asegúrese de que la arquitectura del sistema de archivos deseada para un despliegue de BeeGFS en particular está bien comprendida antes de continuar con la instalación.

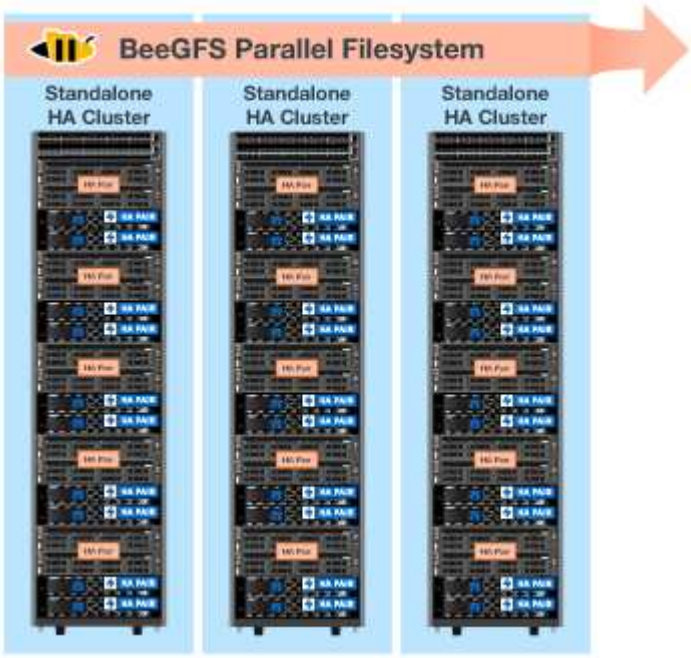
Hardware en rack

Al planificar la instalación, es importante que todo el equipo de cada bloque de construcción esté montado en rack adyacente. La práctica recomendada es que los nodos de archivo estén montados en rack inmediatamente encima de los nodos de bloque en cada bloque básico. Siga la documentación de los modelos de archivo y. "bloque" los nodos que usa mientras instala rieles y hardware en el rack.

Ejemplo de un solo elemento básico:

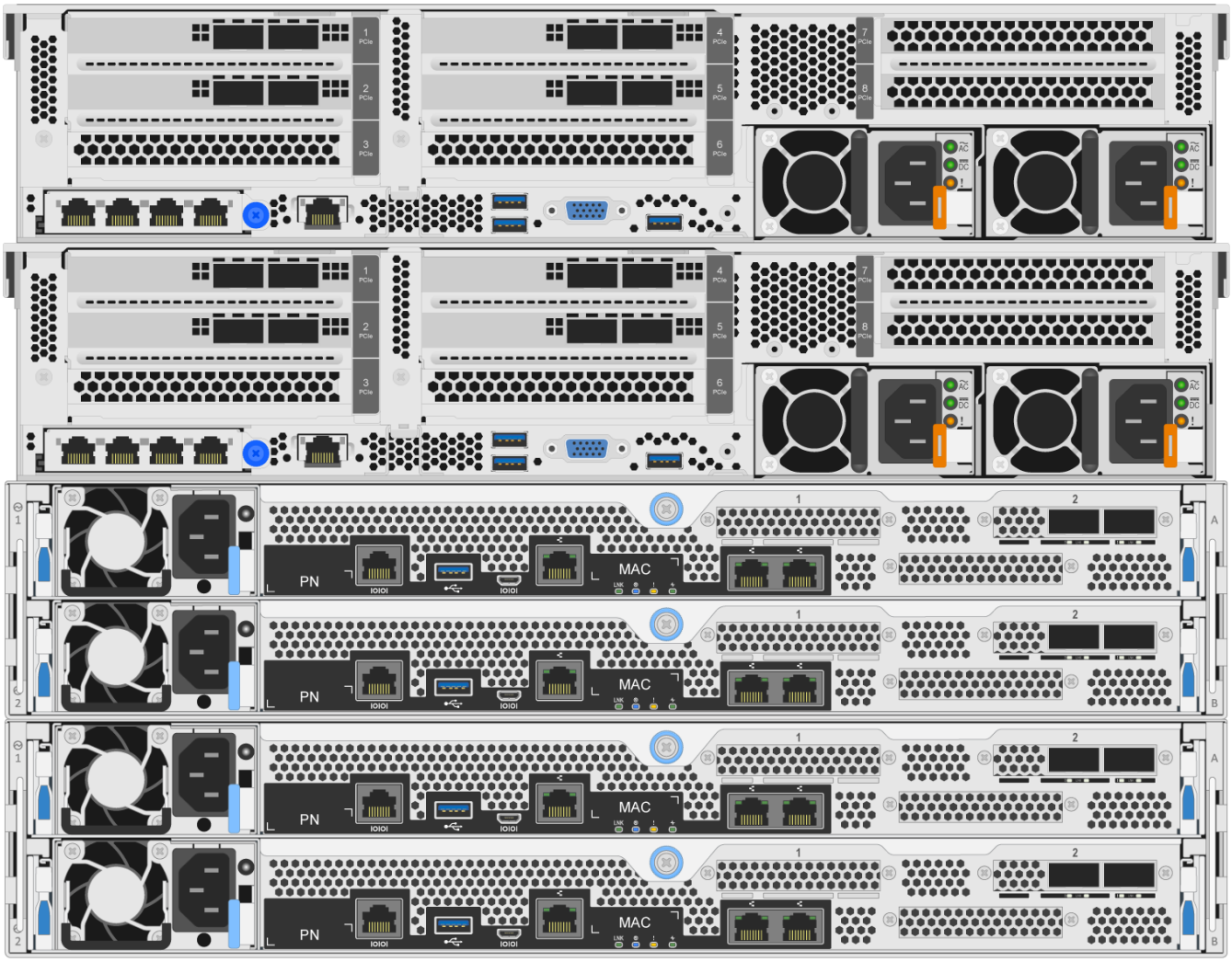


Ejemplo de una instalación de BeeGFS de gran tamaño en la que hay varios elementos básicos en cada clúster de alta disponibilidad y varios clústeres de alta disponibilidad en el sistema de archivos:



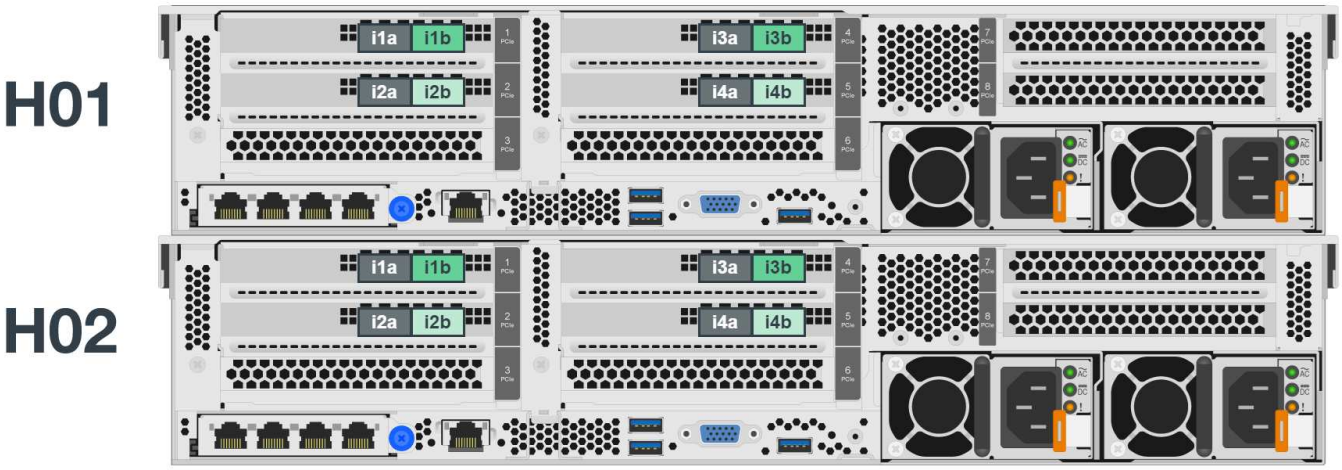
Nodos de archivos de cable y bloques

Normalmente, se conectan directamente los puertos HIC de los nodos de bloque de E-Series al adaptador de canal de host designado (para protocolos InfiniBand) o al adaptador de bus de host (para el canal de fibra y otros protocolos) de los nodos de archivo. La forma exacta de establecer estas conexiones dependerá de la arquitectura del sistema de archivos deseada, aquí hay un ejemplo "Basado en BeeGFS de segunda generación en la arquitectura verificada de NetApp":



Conecte los nodos de archivo a la red de cliente

Cada nodo de archivo tendrá un número de puertos InfiniBand o Ethernet designados para el tráfico del cliente BeeGFS. Dependiendo de la arquitectura que tenga cada nodo de archivo tendrá una o varias conexiones a una red cliente/almacenamiento de alto rendimiento, potencialmente a varios switches para obtener redundancia y un mayor ancho de banda. A continuación se muestra un ejemplo de cableado de cliente mediante switches de red redundantes, donde los puertos resaltados en verde oscuro frente al verde claro se conectan a switches separados:



Conecte la red y la alimentación de la administración

Establezca las conexiones de red necesarias para la red dentro y fuera de banda.

Conecte todas las fuentes de alimentación asegurándose de que cada nodo de archivo y bloque tenga conexiones a varias unidades de distribución de alimentación para obtener redundancia (si está disponible).

Configure los nodos de archivos y bloques

Pasos manuales necesarios para configurar nodos de archivos y bloques antes de ejecutar Ansible.

Nodos de archivos

Configurar el controlador de administración de la placa base (BMC)

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 conoce como controlador XClaridad Lenovo (XCC).

Siga la documentación del proveedor del servidor para habilitar las licencias necesarias para acceder a esta funcionalidad y asegurarse de que el BMC está conectado a la red y configurado de forma adecuada para el acceso remoto.

> Si desea utilizar la cercado basada en BMC con Redfish, asegúrese de que Redfish esté activado y de que se pueda acceder a la interfaz BMC desde el sistema operativo instalado en el nodo de archivo. Es posible que se requiera una configuración especial en el conmutador de red si el BMC y el sistema operativo comparten la misma interfaz de red física.

Ajuste la configuración del sistema

Utilizando la interfaz de configuración del sistema (BIOS/UEFI), asegúrese de que los ajustes se han establecido para maximizar el rendimiento. La configuración exacta y los valores óptimos variarán en función del modelo de servidor que se esté utilizando. Se proporciona orientación para "modelos de nodos de archivos verificados", de lo contrario, consulte la documentación del proveedor del servidor y las mejores prácticas basadas en su modelo.

Instale un sistema operativo

Instale un sistema operativo compatible basado en los requisitos de nodo de archivo que se muestran "aquí". Consulte los pasos adicionales que se indican a continuación según su distribución de Linux.

Red Hat

Utilice RedHat Subscription Manager para registrar y suscribirse al 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=<MAJOR_VERSION>.<MINOR_VERSION>`. Para ver instrucciones, consulte "Cómo registrar y suscribirse a un sistema RHEL" y.. "Cómo limitar las actualizaciones".

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
```

Configure la red de gestión

Configure las interfaces de red necesarias para permitir la administración en banda del sistema operativo. Los pasos exactos dependerán de la distribución y versión específicas de Linux que se esté utilizando.

> Compruebe que SSH esté habilitado y que todas las interfaces de gestión sean accesibles desde el nodo de control de Ansible.

Actualice el firmware de HCA y HBA

Asegúrese de que todos los HBA y HCA están ejecutando las versiones de firmware compatibles enumeradas en "Matriz de interoperabilidad de NetApp"y, si es necesario, actualícelos. Se pueden encontrar recomendaciones adicionales para los adaptadores NVIDIA ConnectX "aquí".

Nodos de bloques

Siga los pasos a. "Póngase en marcha con E-Series" para configurar el puerto de gestión en cada controladora del nodo de bloque y, opcionalmente, establezca el nombre de la cabina de almacenamiento para cada sistema.

> No será necesario realizar ninguna configuración adicional más allá de garantizar que todos los nodos de bloques sean accesibles desde el nodo de control de Ansible. La configuración del sistema restante se aplicará/mantendrá con Ansible.

Configure el nodo de control de Ansible

Configure un nodo de control de Ansible para poner en marcha y gestionar el sistema de archivos.

Descripción general

Un nodo de control de Ansible es una máquina física o virtual Linux que se usa para gestionar el clúster. Debe cumplir los siguientes requisitos:

- Cumpla el "requisitos"rol de alta disponibilidad de BeeGFS, incluidas las versiones instaladas de Ansible, Python y cualquier otro paquete de Python adicional.
- Conozca al funcionario "Requisitos del nodo de control de Ansible" incluye las versiones del sistema operativo.
- Tienen acceso SSH y HTTPS a todos los nodos de archivos y bloques.

Se pueden encontrar pasos de instalación detallados"aquí".

Defina el sistema de archivos BeeGFS

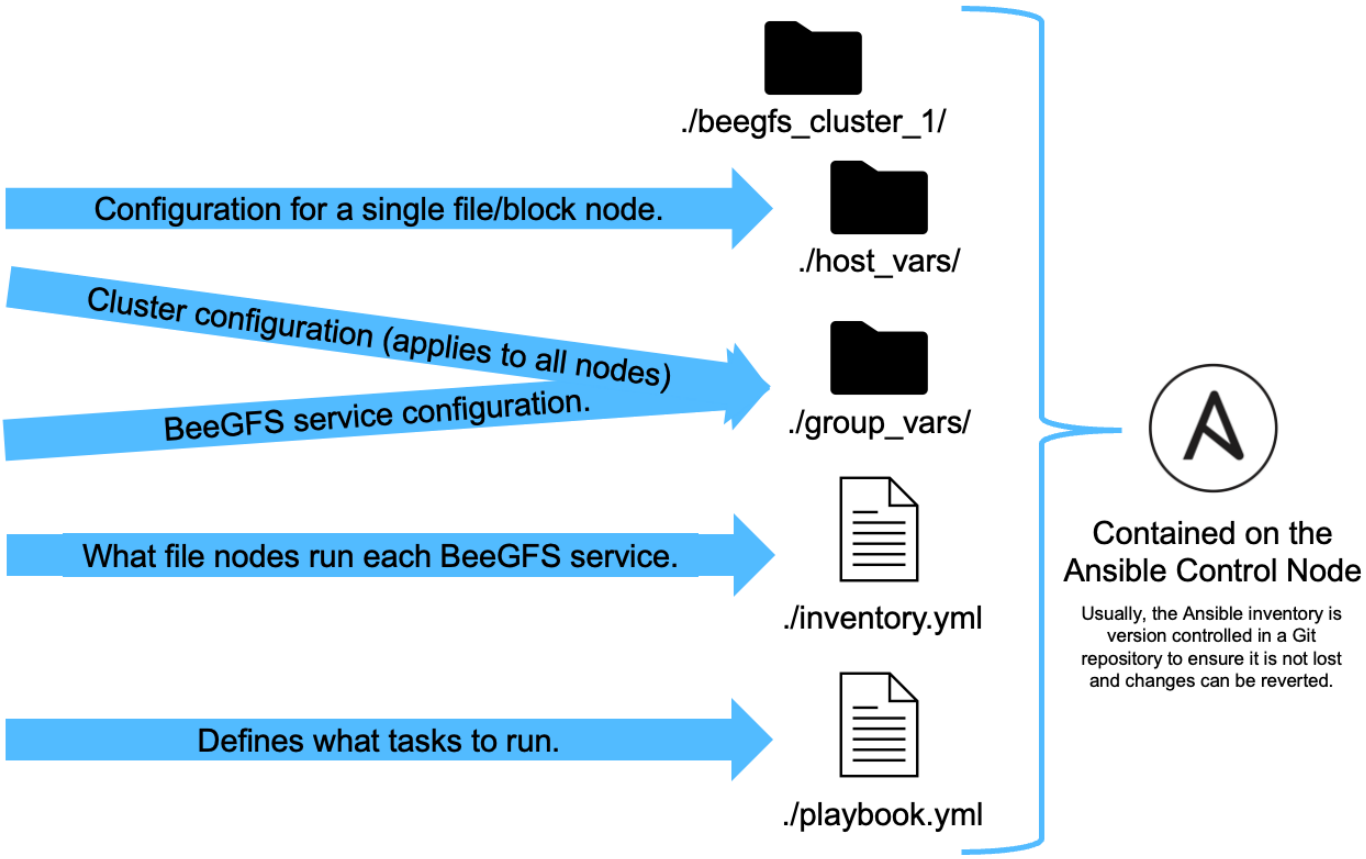
Descripción general del inventario de Ansible

El inventario de Ansible es un conjunto de archivos de configuración que definen el clúster de alta disponibilidad de BeeGFS deseado.

Descripción general

Se recomienda seguir las prácticas de Ansible estándar para organizar el "inventario", incluido el uso de "subdirectorios/archivos" en lugar de almacenar todo el inventario en un archivo.

El inventario de Ansible para un único clúster de alta disponibilidad de BeeGFS está organizado de la siguiente forma:



Dado que un único sistema de archivos BeeGFS puede abarcar varios clústeres de alta disponibilidad, es posible que las instalaciones de gran tamaño tengan varios inventarios de Ansible. Por lo general, no se recomienda intentar definir varios clústeres de alta disponibilidad como un único inventario de Ansible para evitar problemas.

Pasos

1. En el nodo de control de Ansible, cree un directorio vacío que contendrá el inventario de Ansible para el clúster de BeeGFS que desea implementar.
   a. Si su sistema de archivos contendrá en algún momento varios clústeres de alta disponibilidad, se recomienda crear primero un directorio para el sistema de archivos y, a continuación, subdirectorios para el inventario que represente cada clúster de alta disponibilidad. Por ejemplo:

```
beegfs_file_system_1/
    beegfs_cluster_1/
    beegfs_cluster_2/
    beegfs_cluster_N/
```

2. En el directorio que contiene el inventario del clúster de alta disponibilidad que desea implementar, cree dos directorios `group_vars` y.. `host_vars` y dos archivos `inventory.yml` y.. `playbook.yml`.

En las siguientes secciones se describe la definición del contenido de cada uno de estos archivos.

Planifique el sistema de archivos

Planifique la puesta en marcha del sistema de archivos antes de crear el inventario de Ansible.

Descripción general

Antes de implementar el sistema de archivos, debe definir qué direcciones IP, puertos y otra configuración serán necesarias para todos los nodos de archivos, nodos de bloques y servicios BeeGFS que se ejecuten en el clúster. Aunque la configuración exacta variará en función de la arquitectura del clúster, esta sección define las prácticas recomendadas y los pasos a seguir que son aplicables.

Pasos

1. Si utiliza un protocolo de almacenamiento basado en IP (como Iser, iSCSI, NVMe/IB o NVMe/roce) para conectar nodos de archivos a nodos de bloques, rellene la siguiente hoja de datos para cada elemento básico. Cada conexión directa en un único bloque de creación debería tener una subred única y no debería haber superposición con subredes utilizadas para la conectividad cliente-servidor.

Nodo de archivo	Puerto IB	Dirección IP	Nodo de bloques	Puerto IB	IP física	Virtual IP (solo para EF600 con HDR IB)
<HOSTNAME>	<PORT>	<IP/SUBNET>	<HOSTNAME>	<PORT>	<IP/SUBNET>	<IP/SUBNET>

Si los nodos de archivo y bloque de cada bloque de creación están conectados directamente, a menudo puede reutilizar las mismas IP/esquema para varios bloques de creación.

2. Independientemente de si utiliza InfiniBand o RDMA sobre Ethernet convergente (roce) para la red de almacenamiento, rellene la siguiente hoja de datos para determinar los rangos de IP que se usarán para los servicios de clúster de alta disponibilidad, los servicios de archivos BeeGFS y los clientes para comunicarse:

Específico	Puerto InfiniBand	Dirección IP o rango
IP(s) de clúster de BeeGFS	<INTERFACE(s)>	<RANGE>
Gestión de BeeGFS	<INTERFACE(s)>	<IP(s)>
Metadatos de BeeGFS	<INTERFACE(s)>	<RANGE>
Almacenamiento de BeeGFS	<INTERFACE(s)>	<RANGE>
Clientes BeeGFS	<INTERFACE(s)>	<RANGE>

   a. Si utiliza una sola subred IP, solo será necesaria una hoja de datos; de lo contrario, rellene también una hoja de cálculo para la segunda subred.

3. En función de lo anterior, para cada bloque de creación del clúster, rellene la siguiente hoja de trabajo que define qué servicios de BeeGFS se ejecutará. Para cada servicio, especifique los nodos de archivo preferidos/secundarios, el puerto de red, las IP flotantes, la asignación de zonas NUMA (si es necesario) y qué nodos de bloque se usarán para sus destinos. Consulte las siguientes directrices al rellenar la hoja de trabajo:

   a. Especifique los servicios BeeGFS como cualquiera de los dos `mgmt.yml`, `meta_<ID>.yml`, o. `storage_<ID>.yml` Donde ID representa un número único en todos los servicios BeeGFS de ese tipo en este sistema de archivos. Esta convención simplificará la referencia a esta hoja de trabajo en secciones posteriores mientras crea archivos para configurar cada servicio.

   b. Los puertos para los servicios BeeGFS sólo deben ser únicos en un bloque de construcción en particular. Asegúrese de que los servicios con el mismo número de puerto no pueden ejecutarse nunca en el mismo nodo de archivo para evitar conflictos de puertos.

   c. Si los servicios necesarios pueden utilizar volúmenes de más de un nodo de bloque o un pool de almacenamiento (y no todos los volúmenes deben ser propiedad de la misma controladora). Múltiples servicios también pueden compartir la misma configuración de nodo de bloque o pool de almacenamiento (se definirán los volúmenes individuales en una sección posterior).

Servicio BeeGFS (nombre de archivo)	Nodos de archivos	Puerto	IP flotantes	Zona NUMA	Nodo de bloques	Del banco de almacenamiento	Controladora propietaria
<SERVICE TYPE>_<ID>.yml	<PREFERRED FILE NODE> <SECONDARY FILE NODE(s)>	<PORT>	<INTERFACE>:<IP/SUBNET> <INTERFACE>:<IP/SUBNET>	<NUMA NODE/ZONE>	<BLOCK NODE>	<STORAGE POOL/VOLUME GROUP>	<A OR B>

Para obtener más información sobre las convenciones estándar, las mejores prácticas y las hojas de trabajo completadas de ejemplo, consulte "mejores prácticas""Defina los bloques de creación de BeeGFS"las secciones y de la arquitectura verificada de BeeGFS en NetApp.

Defina los nodos de archivo y bloque

Configurar nodos de archivos individuales

Especifique la configuración de los nodos de archivos individuales con variables de host (host_var).

Descripción general

Esta sección recorre la relleno de un `host_vars/<FILE_NODE_HOSTNAME>.yml` archivo para cada nodo de archivo del clúster. Estos archivos sólo deben contener una configuración exclusiva de un nodo de archivo concreto. Esto incluye normalmente:

- Definición de la IP o el nombre de host que Ansible debe usar para conectarse al nodo.
- Configurar interfaces adicionales e IP de clúster utilizadas para los servicios de clúster de alta disponibilidad (Pacemaker y Corosync) para comunicarse con otros nodos de archivo. De forma predeterminada, estos servicios utilizan la misma red que la interfaz de gestión, pero deberían estar disponibles interfaces adicionales para la redundancia. La práctica común es definir IP adicionales en la red de almacenamiento, lo que evita la necesidad de un clúster o una red de gestión adicionales.
  - El rendimiento de cualquier red utilizada para la comunicación del clúster no es crítico en cuanto al rendimiento del sistema de archivos. Con la configuración de clúster predeterminada, por lo general, al menos una red de 1GB GB/s proporcionará suficiente rendimiento para las operaciones de clúster, como la sincronización de estados de nodo y la coordinación de cambios de estado de recursos de clúster. Las redes lentas/ocupadas pueden hacer que los cambios en el estado de los recursos tarden más de lo habitual y, en casos extremos, podrían resultar en que los nodos se expulsen del clúster si no pueden enviar latidos en un período de tiempo razonable.
- Configurar las interfaces utilizadas para conectarse a los nodos de bloques sobre el protocolo deseado (por ejemplo, iSCSI/Iser, NVMe/IB, NVMe/roce, FCP, etc.)

Pasos

Haciendo referencia al esquema de direcciones IP definido en la "Planifique el sistema de archivos" sección, para cada nodo de archivo del cluster cree un archivo `host_vars/<FILE_NODE_HOSTNAME>/yml` y rellénelo de la siguiente manera:

1. En la parte superior, especifique la IP o el nombre de host que Ansible debe usar a SSH del nodo y gestiónelo:

```
ansible_host: "<MANAGEMENT_IP>"
```

2. Configure las IP adicionales que se puedan usar para el tráfico del clúster:
   a. Si el tipo de red es "InfiniBand (uso de IPoIB)":

```
eseries_ipoib_interfaces:
- name: <INTERFACE>  # Example: ib0 or i1b
  address: <IP/SUBNET> # Example: 100.127.100.1/16
- name: <INTERFACE>  # Additional interfaces as needed.
  address: <IP/SUBNET>
```

b. Si el tipo de red es "RDMA sobre Ethernet convergente (roce)":

```
eseries_roce_interfaces:
- name: <INTERFACE>  # Example: eth0.
  address: <IP/SUBNET> # Example: 100.127.100.1/16
- name: <INTERFACE>  # Additional interfaces as needed.
  address: <IP/SUBNET>
```

c. Si el tipo de red es "Ethernet (solo TCP, sin RDMA)":

```
eseries_ip_interfaces:
- name: <INTERFACE>  # Example: eth0.
  address: <IP/SUBNET> # Example: 100.127.100.1/16
- name: <INTERFACE>  # Additional interfaces as needed.
  address: <IP/SUBNET>
```

3. Indique qué IP se deben utilizar para el tráfico del clúster con las IP preferidas más alta:

```
beegfs_ha_cluster_node_ips:
- <MANAGEMENT_IP> # Including the management IP is typically but not
required.
- <IP_ADDRESS>    # Ex: 100.127.100.1
- <IP_ADDRESS>    # Additional IPs as needed.
```

> Los IPS configurados en el paso dos no se utilizarán como IP de clúster a menos que estén incluidos en el `beegfs_ha_cluster_node_ips` lista. Esto le permite configurar IP/interfaces adicionales con Ansible que pueden utilizarse para otros fines si así lo desea.

4. Si el nodo de archivo tiene que comunicarse con los nodos de bloque a través de un protocolo basado en IP, se deberán configurar las IP en la interfaz adecuada y con todos los paquetes necesarios para instalar y configurar ese protocolo.

   a. Si se utiliza "ISCSI":

```
eseries_iscsi_interfaces:
- name: <INTERFACE>  # Example: eth0.
  address: <IP/SUBNET> # Example: 100.127.100.1/16
```

b. Si se utiliza "Iser":

```
eseries_ib_iser_interfaces:
- name: <INTERFACE>  # Example: ib0.
  address: <IP/SUBNET> # Example: 100.127.100.1/16
  configure: true # If the file node is directly connected to the
block node set to true to setup OpenSM.
```

c. Si se utiliza "NVMe/IB":

```
eseries_nvme_ib_interfaces:
- name: <INTERFACE>  # Example: ib0.
  address: <IP/SUBNET> # Example: 100.127.100.1/16
  configure: true # If the file node is directly connected to the
block node set to true to setup OpenSM.
```

d. Si se utiliza "NVMe/roce":

```
eseries_nvme_roce_interfaces:
- name: <INTERFACE>  # Example: eth0.
  address: <IP/SUBNET> # Example: 100.127.100.1/16
```

e. Otros protocolos:

   i. Si se utiliza "NVMe/FC", no es necesario configurar interfaces individuales. La implementación del clúster BeeGFS detectará automáticamente los requisitos de protocolo e instalará/configurará según sea necesario. Si utiliza una estructura para conectar nodos de archivos y bloques, asegúrese de que los switches se dividen correctamente siguiendo las prácticas recomendadas de NetApp y del proveedor del switch.

   ii. El uso de FCP o SAS no requiere la instalación ni la configuración de software adicional. Si utiliza FCP, asegúrese de que los switches se dividen correctamente a continuación "NetApp" y las prácticas recomendadas de su proveedor del switch.

   iii. No se recomienda el uso de SRP IB en este momento. Utilice NVMe/IB o Iser en función de lo que admita su nodo de bloque E-Series.

Haga clic en "aquí" para obtener un ejemplo de un archivo de inventario completo que representa un solo nodo de archivo.

Avanzado: Alternar los adaptadores VPI NVIDIA ConnectX entre Ethernet y modo InfiniBand

Los adaptadores NVIDIA ConnectX-Virtual Protocol Interconnect® (VPI) admiten InfiniBand y Ethernet como capa de transporte. El cambio entre modos no se negocia automáticamente y debe configurarse mediante la <code>mstconfig</code> herramienta incluida en <code>mstflint</code>, un paquete de código abierto que forma parte de "Herramientas de Firmare de NVIDIA (MFT)". El cambio del modo de los adaptadores solo debe realizarse una vez. Esto se puede hacer manualmente, o incluir en el inventario de Ansible como parte de cualquier interfaz configurada usando la <code>eseries-[ib|ib_iser|ipoib|nvme_ib|nvme_roce|roce]_interfaces:</code> sección del inventario, para que se active/aplique automáticamente.

Por ejemplo, para cambiar una interfaz actual en modo InfiniBand a Ethernet, se puede utilizar para roce:

1. Especifique para cada interfaz que desee configurar `mstconfig` como una asignación (o diccionario) que especifica `LINK_TYPE_P<N>` donde `<N>` Viene determinado por el número de puerto de HCA de la interfaz. La `<N>` el valor se puede determinar ejecutando `grep PCI_SLOT_NAME /sys/class/net/<INTERFACE_NAME>/device/uevent` Y agregando 1 al último número desde el nombre de la ranura PCI y convirtiendo a decimal.
   a. Por ejemplo dado `PCI_SLOT_NAME=0000:2f:00.2` (2 + 1 → puerto HCA 3) → `LINK_TYPE_P3: eth`:

```
eseries_roce_interfaces:
- name: <INTERFACE>
  address: <IP/SUBNET>
  mstconfig:
    LINK_TYPE_P3: eth
```

Para obtener información adicional, consulte "Documentación de la colección de hosts E-Series de NetApp" para el tipo/protocolo de interfaz que utiliza.

Configure nodos de bloques individuales

Especifique la configuración de los nodos de bloque individuales con variables de host (host_var).

Descripción general

Esta sección recorre la relleno de un `host_vars/<BLOCK_NODE_HOSTNAME>.yml` archivo de cada nodo de bloque del clúster. Estos archivos sólo deben contener una configuración exclusiva de un nodo de bloque determinado. Esto incluye normalmente:

- El nombre del sistema (como se muestra en System Manager).
- La URL de HTTPS para una de las controladoras (se utiliza para gestionar el sistema mediante su API DE REST).
- Qué nodos de archivo de protocolo de almacenamiento utilizan para conectarse a este nodo de bloque.
- Configurar los puertos de tarjeta de interfaz del host (HIC), como las direcciones IP (si son necesarias).

Pasos

Haciendo referencia al esquema de direcciones IP definido en la "Planifique el sistema de archivos" sección, para cada nodo de bloque del cluster cree un archivo `host_vars/<BLOCK_NODE_HOSTNAME>/yml` y rellénelo de la siguiente manera:

1. En la parte superior, especifique el nombre del sistema y la URL de HTTPS para una de las controladoras:

```
eseries_system_name: <SYSTEM_NAME>
eseries_system_api_url:
https://<MANAGEMENT_HOSTNAME_OR_IP>:8443/devmgr/v2/
```

2. Seleccione la "protocolo" los nodos de archivo utilizarán para conectarse a este nodo de bloque:
   a. Protocolos compatibles: `auto`, `iscsi`, `fc`, `sas`, `ib_srp`, `ib_iser`, `nvme_ib`, `nvme_fc`, `nvme_roce`.

```
eseries_initiator_protocol: <PROTOCOL>
```

3. Según el protocolo en uso, los puertos HIC pueden necesitar una configuración adicional. Si es necesario, se debe definir la configuración de puertos de HIC para que la entrada superior de la configuración de cada controladora se corresponda con el puerto físico más a la izquierda de cada controladora y el puerto inferior al puerto que se encuentra en el extremo derecho. Todos los puertos requieren una configuración válida incluso si no están en uso actualmente.

   Consulte también la siguiente sección si utiliza InfiniBand HDR (200 GB) o roce de 200 GB con nodos de bloque EF600.

   a. Para iSCSI:

```
eseries_controller_iscsi_port:
  controller_a:         # Ordered list of controller A channel
definition.
    - state:            # Whether the port should be enabled.
Choices: enabled, disabled
      config_method:    # Port configuration method Choices: static,
dhcp
      address:          # Port IPv4 address
      gateway:          # Port IPv4 gateway
      subnet_mask:      # Port IPv4 subnet_mask
      mtu:              # Port IPv4 mtu
    - (...)             # Additional ports as needed.
  controller_b:         # Ordered list of controller B channel
definition.
    - (...)             # Same as controller A but for controller B

# Alternatively the following common port configuration can be
defined for all ports and omitted above:
eseries_controller_iscsi_port_state: enabled         # Generally
specifies whether a controller port definition should be applied
Choices: enabled, disabled
eseries_controller_iscsi_port_config_method: dhcp    # General port
configuration method definition for both controllers. Choices:
static, dhcp
eseries_controller_iscsi_port_gateway:               # General port
IPv4 gateway for both controllers.
eseries_controller_iscsi_port_subnet_mask:           # General port
IPv4 subnet mask for both controllers.
eseries_controller_iscsi_port_mtu: 9000              # General port
maximum transfer units (MTU) for both controllers. Any value greater
than 1500 (bytes).
```

b. Para Iser:

```
eseries_controller_ib_iser_port:
  controller_a:    # Ordered list of controller A channel address
definition.
    -              # Port IPv4 address for channel 1
    - (...)        # So on and so forth
  controller_b:    # Ordered list of controller B channel address
definition.
```

c. Para NVMe/IB:

```
eseries_controller_nvme_ib_port:
  controller_a:    # Ordered list of controller A channel address
definition.
    -              # Port IPv4 address for channel 1
    - (...)        # So on and so forth
  controller_b:    # Ordered list of controller B channel address
definition.
```

d. Para NVMe/roce:

```
eseries_controller_nvme_roce_port:
  controller_a:         # Ordered list of controller A channel
definition.
    - state:            # Whether the port should be enabled.
      config_method:    # Port configuration method Choices: static,
dhcp
      address:          # Port IPv4 address
      subnet_mask:      # Port IPv4 subnet_mask
      gateway:          # Port IPv4 gateway
      mtu:              # Port IPv4 mtu
      speed:            # Port IPv4 speed
  controller_b:         # Ordered list of controller B channel
definition.
    - (...)             # Same as controller A but for controller B

# Alternatively the following common port configuration can be
defined for all ports and omitted above:
eseries_controller_nvme_roce_port_state: enabled         # Generally
specifies whether a controller port definition should be applied
Choices: enabled, disabled
eseries_controller_nvme_roce_port_config_method: dhcp    # General
port configuration method definition for both controllers. Choices:
static, dhcp
eseries_controller_nvme_roce_port_gateway:               # General
port IPv4 gateway for both controllers.
eseries_controller_nvme_roce_port_subnet_mask:           # General
port IPv4 subnet mask for both controllers.
eseries_controller_nvme_roce_port_mtu: 4200              # General
port maximum transfer units (MTU). Any value greater than 1500
(bytes).
eseries_controller_nvme_roce_port_speed: auto            # General
interface speed. Value must be a supported speed or auto for
automatically negotiating the speed with the port.
```

e. Los protocolos FC y SAS no requieren configuración adicional. SRP no se recomienda correctamente.

Para obtener opciones adicionales para configurar los puertos HIC y los protocolos de host, incluida la capacidad de configurar CHAP iSCSI, consulte la "documentación" Incluido con la colección SANtricity. Tenga en cuenta que al implementar BeeGFS, el pool de almacenamiento, la configuración de volumen y otros aspectos del aprovisionamiento del almacenamiento se configurarán en otra parte y no se deberán definir en este archivo.

Haga clic en "aquí" para obtener un ejemplo de un archivo de inventario completo que representa un solo nodo de bloque.

Mediante InfiniBand HDR (200 GB) o roce de 200 GB con nodos de bloque de EF600 de NetApp:

Para utilizar InfiniBand HDR (200 GB) con EF600, se debe configurar una segunda IP "virtual" para cada puerto físico. A continuación se muestra un ejemplo de la forma correcta de configurar un EF600 equipado con la HIC HDR InfiniBand de doble puerto:

```
eseries_controller_nvme_ib_port:
  controller_a:
    - 192.168.1.101   # Port 2a (virtual)
    - 192.168.2.101   # Port 2b (virtual)
    - 192.168.1.100   # Port 2a (physical)
    - 192.168.2.100   # Port 2b (physical)
  controller_b:
    - 192.168.3.101   # Port 2a (virtual)
    - 192.168.4.101   # Port 2b (virtual)
    - 192.168.3.100   # Port 2a (physical)
    - 192.168.4.100   # Port 2b (physical)
```

Especifique la configuración de nodos de archivos comunes

Especifique la configuración de nodos de archivos comunes mediante variables de grupo (group_var).

Descripción general

La configuración que debería Apple para todos los nodos de archivo se define en `group_vars/ha_cluster.yml`. Normalmente incluye:

- Información detallada sobre cómo conectarse e iniciar sesión en cada nodo de archivo.
- Configuración de red común.
- Si se permiten reinicios automáticos.
- Cómo deben configurarse los estados de firewall y selinux.
- Configuración de clústeres, incluidas las alertas y las cercas.
- Ajuste del rendimiento.
- Configuración de servicio de BeeGFS común.

> Las opciones establecidas en este archivo también pueden definirse en nodos de archivos individuales; por ejemplo, si se están usando modelos de hardware mixtos o tiene contraseñas diferentes para cada nodo. La configuración en nodos de archivo individuales tendrá prioridad sobre la configuración de este archivo.

Pasos

Cree el archivo `group_vars/ha_cluster.yml` y rellenarlo de la siguiente manera:

1. Indique cómo debe autenticarse el nodo de Ansible Control con los hosts remotos:

```
ansible_ssh_user: root
ansible_become_password: <PASSWORD>
```

> Especialmente en entornos de producción, no almacene contraseñas en texto sin formato. En su lugar, use Ansible Vault (consulte "Cifrado de contenido con Ansible Vault") o el `--ask-become-pass` al ejecutar el libro de estrategia. Si la `ansible_ssh_user` es ya el usuario raíz. puede omitir de forma opcional el `ansible_become_password`.

2. Si está configurando IP estáticas en interfaces ethernet o InfiniBand (por ejemplo, IP de clúster) y varias interfaces se encuentran en la misma subred IP (por ejemplo, si ib0 está usando 192.168.1.10/24 e ib1 está usando 192.168.1.11/24), Para que el soporte multihost funcione correctamente, se deben configurar reglas y tablas de enrutamiento IP adicionales. Sólo tiene que activar el enlace de configuración de la interfaz de red proporcionado de la siguiente forma:

```
eseries_ip_default_hook_templates:
  - 99-multihoming.j2
```

3. Al poner en marcha el clúster, según el protocolo de almacenamiento, puede que sean necesarios el reinicio de los nodos para facilitar la detección de dispositivos de bloques remotos (volúmenes de E-Series) o aplicar otros aspectos de la configuración. De forma predeterminada, los nodos se preguntará antes de reiniciar, pero puede permitir que los nodos se reinicien automáticamente especificando lo siguiente:

```
eseries_common_allow_host_reboot: true
```

   a. De forma predeterminada, después de un reinicio, para asegurarse de que los dispositivos de bloque y otros servicios estén listos, Ansible esperará hasta el sistema `default.target` se alcanza antes de continuar con la implementación. En algunos casos, cuando se utiliza NVMe/IB, es posible que este no sea el tiempo suficiente para inicializar, detectar y conectarse a dispositivos remotos. Esto puede provocar que la implementación automatizada continúe prematuramente y falle. Para evitar esto cuando se usa NVMe/IB, también se debe definir lo siguiente:

```
eseries_common_reboot_test_command: "! systemctl status
eseries_nvme_ib.service || systemctl --state=exited | grep
eseries_nvme_ib.service"
```

4. Para comunicarse con los servicios de clúster de BeeGFS y ha se necesitan varios puertos de firewall. A menos que desee configurar manualmente el firmwwall (no se recomienda), especifique lo siguiente para crear zonas de firewall necesarias y abrir puertos automáticamente:

```
beegfs_ha_firewall_configure: True
```

5. En este momento, SELinux no es compatible y se recomienda que el estado se configure como desactivado para evitar conflictos (especialmente cuando RDMA está en uso). Establezca lo siguiente para asegurarse de que SELinux esté desactivado:

```
eseries_beegfs_ha_disable_selinux: True
eseries_selinux_state: disabled
```

6. Configure la autenticación de modo que los nodos de archivo puedan comunicarse, ajustando los valores predeterminados según sea necesario según las directivas de su organización:

```
beegfs_ha_cluster_name: hacluster                  # BeeGFS HA cluster
name.
beegfs_ha_cluster_username: hacluster              # BeeGFS HA cluster
username.
beegfs_ha_cluster_password: hapassword             # BeeGFS HA cluster
username's password.
beegfs_ha_cluster_password_sha512_salt: randomSalt # BeeGFS HA cluster
username's password salt.
```

7. En función de la "Planifique el sistema de archivos" sección, especifique la IP de administración de BeeGFS para este sistema de archivos:

```
beegfs_ha_mgmtd_floating_ip: <IP ADDRESS>
```

> 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.

8. Habilite las alertas de correo electrónico si lo desea:

```
beegfs_ha_enable_alerts: True
# E-mail recipient list for notifications when BeeGFS HA resources
change or fail.
beegfs_ha_alert_email_list: ["<EMAIL>"]
# This dictionary is used to configure postfix service
(/etc/postfix/main.cf) which is required to set email alerts.
beegfs_ha_alert_conf_ha_group_options:
      # This parameter specifies the local internet domain name. This is
optional when the cluster nodes have fully qualified hostnames (i.e.
host.example.com)
      mydomain: <MY_DOMAIN>
beegfs_ha_alert_verbosity: 3
#  1) high-level node activity
#  3) high-level node activity + fencing action information + resources
(filter on X-monitor)
#  5) high-level node activity + fencing action information + resources
```

9. Se recomienda encarecidamente habilitar la delimitación; de lo contrario, se puede bloquear que los servicios se inicien en nodos secundarios cuando se produzca un error en el nodo principal.
   a. Active la delimitación de forma global especificando lo siguiente:

```
beegfs_ha_cluster_crm_config_options:
  stonith-enabled: True
```

      i. Nota Cualquier compatible "propiedad del clúster" también se puede especificar aquí si es necesario. No suele ser necesario ajustar estos ajustes, ya que el papel BeeGFS HA se entrega con una serie de pruebas bien probadas "valores predeterminados".

   b. A continuación, seleccione y configure un agente de cercado:
      i. OPCIÓN 1: Para habilitar la cercado mediante unidades de distribución de energía (PDU) APC:

```
beegfs_ha_fencing_agents:
  fence_apc:
    - ipaddr: <PDU_IP_ADDRESS>
      login: <PDU_USERNAME>
      passwd: <PDU_PASSWORD>
      pcmk_host_map:
"<HOSTNAME>:<PDU_PORT>,<PDU_PORT>;<HOSTNAME>:<PDU_PORT>,<PDU_PORT>
"
```

      ii. OPCIÓN 2: Para habilitar la esgrima mediante las API Redfish proporcionadas por Lenovo XCC (y otros BMCs):

```
redfish: &redfish
  username: <BMC_USERNAME>
  password: <BMC_PASSWORD>
  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
```

iii. Para obtener más información sobre la configuración de otros agentes de cercado, consulte la "Documentación de redhat".

10. El rol de ha de BeeGFS puede aplicar muchos parámetros de ajuste diferentes para ayudar a optimizar aún más el rendimiento. Entre ellos se incluyen la optimización de la utilización de la memoria del núcleo y la E/S del dispositivo en bloque, entre otros parámetros. El rol se incluye con un conjunto razonable de "valores predeterminados" basado en pruebas con nodos de bloque NetApp E-Series, pero de forma predeterminada estos no se aplican a menos que especifique:

```
beegfs_ha_enable_performance_tuning: True
```

a. Si es necesario, también especifique aquí cualquier cambio en el ajuste del rendimiento predeterminado. Consulte la documentación completa "parámetros de ajuste del rendimiento" para obtener más información.

11. Para garantizar que las direcciones IP flotantes (a veces conocidas como interfaces lógicas) utilizadas para los servicios BeeGFS puedan conmutar por error entre nodos de archivos, todas las interfaces de red deben tener un nombre coherente. De forma predeterminada, el kernel genera nombres de interfaz de red, lo cual no garantiza la generación de nombres coherentes, incluso en modelos de servidor idénticos con adaptadores de red instalados en las mismas ranuras PCIe. Esto también es útil cuando se crean inventarios antes de que el equipo se despliegue y se conozcan los nombres de las interfaces generadas. Para garantizar nombres de dispositivos coherentes, basados en un diagrama de bloque del servidor o. `lshw -class network -businfo` Output, especifique la asignación de dirección PCIe a interfaz lógica deseada del siguiente modo:

a. Para interfaces de red InfiniBand (IPoIB):

```
eseries_ipoib_udev_rules:
  "<PCIe ADDRESS>": <NAME> # Ex: 0000:01:00.0: i1a
```

b. Para interfaces de red Ethernet:

```
eseries_ip_udev_rules:
  "<PCIe ADDRESS>": <NAME> # Ex: 0000:01:00.0: e1a
```

> Para evitar conflictos cuando se cambia el nombre de las interfaces (evitando que se le cambie el nombre), no debe utilizar ningún nombre predeterminado potencial como eth0, ens9f0, ib0 o ibs4f0. Una convención de nomenclatura común consiste en usar "e" o "i" para Ethernet o InfiniBand, seguido del número de ranura PCIe y una letra para indicar el puerto. Por ejemplo, el segundo puerto de un adaptador InfiniBand instalado en la ranura 3 sería: I3b.

> Si va a utilizar un modelo de nodo de archivos verificado, haga clic en "aquí" Asignaciones de puerto lógico a dirección PCIe de ejemplo.

12. Opcionalmente, especifique la configuración que debe aplicarse a todos los servicios de BeeGFS del clúster. Se pueden encontrar valores de configuración por defecto "aquí"y la configuración por servicio se especifica en otro lugar:
    a. Servicio de gestión de BeeGFS:

    ```
    beegfs_ha_beegfs_mgmtd_conf_ha_group_options:
      <OPTION>: <VALUE>
    ```

    b. Servicios de metadatos BeeGFS:

    ```
    beegfs_ha_beegfs_meta_conf_ha_group_options:
      <OPTION>: <VALUE>
    ```

    c. Servicios de almacenamiento de BeeGFS:

    ```
    beegfs_ha_beegfs_storage_conf_ha_group_options:
      <OPTION>: <VALUE>
    ```

13. A partir de BeeGFS 7.2.7 y 7.3.1 "autenticación de conexión" se debe configurar o deshabilitar explícitamente. Hay algunas formas de configurar esto con la puesta en marcha basada en Ansible:
    a. De forma predeterminada, la implementación configurará automáticamente la autenticación de conexión y generará un `connauthfile` Se distribuirá a todos los nodos de archivos y se utilizará con los servicios BeeGFS. Este archivo también se colocará/mantendrá en el nodo de control Ansible en `<INVENTORY>/files/beegfs/<sysMgmtdHost>_connAuthFile` donde se debe mantener (de forma segura) para reutilizarlo con clientes que necesiten acceder a este sistema de archivos.
        i. Para generar una nueva clave, especifique `-e "beegfs_ha_conn_auth_force_new=True` Al ejecutar el libro de estrategia de Ansible. Nota esto se ignora si un `beegfs_ha_conn_auth_secret` está definido.
        ii. Para opciones avanzadas, consulte la lista completa de valores predeterminados incluidos con el "Rol de BeeGFS ha".

b. Se puede utilizar un secreto personalizado definiendo lo siguiente en `ha_cluster.yml`:

```
beegfs_ha_conn_auth_secret: <SECRET>
```

c. La autenticación de conexión se puede deshabilitar completamente (NO se recomienda):

```
beegfs_ha_conn_auth_enabled: false
```

Haga clic en "aquí" para obtener un ejemplo de un archivo de inventario completo que representa la configuración común de nodos de archivos.

Usar InfiniBand HDR (200 GB) con nodos de bloque de EF600 de NetApp:

Para utilizar InfiniBand HDR (200 GB) con EF600, el administrador de subredes debe admitir la virtualización. Si los nodos de archivos y bloques se conectan mediante un switch, deberá habilitarse en el administrador de subredes de la estructura general.

Si los nodos de bloques y archivos se conectan directamente mediante InfiniBand, `opensm` se debe configurar una instancia de en cada nodo de archivo para cada interfaz conectada directamente a un nodo de bloque. Esto se hace especificando `configure: true` cuándo "configurar las interfaces de almacenamiento del nodo de archivo".

Actualmente, la versión de bandeja de entrada de `opensm` incluida con distribuciones de Linux compatibles no admite la virtualización. En su lugar, es necesario instalar y configurar la versión de `opensm` desde la distribución empresarial de OpenFabrics (OFED) de NVIDIA. A pesar de que todavía se admite la puesta en marcha con Ansible, se requieren algunos pasos adicionales:

1. Utilizando curl o la herramienta que desee, descargue los paquetes para la versión de OpenSM enumerados en la "requisitos tecnológicos" sección desde el sitio web de NVIDIA al `<INVENTORY>/packages/` directorio. Por ejemplo:

```
curl -o packages/opensm-libs-5.17.2.MLNX20240610.dc7c2998-
0.1.2310322.x86_64.rpm
https://linux.mellanox.com/public/repo/mlnx_ofed/23.10-
3.2.2.0/rhel9.3/x86_64/opensm-libs-5.17.2.MLNX20240610.dc7c2998-
0.1.2310322.x86_64.rpm

curl -o packages/opensm-5.17.2.MLNX20240610.dc7c2998-
0.1.2310322.x86_64.rpm
https://linux.mellanox.com/public/repo/mlnx_ofed/23.10-
3.2.2.0/rhel9.3/x86_64/opensm-5.17.2.MLNX20240610.dc7c2998-
0.1.2310322.x86_64.rpm
```

2. Inferior `group_vars/ha_cluster.yml` defina la siguiente configuración:

```
### OpenSM package and configuration information
eseries_ib_opensm_allow_upgrades: true
eseries_ib_opensm_skip_package_validation: true
eseries_ib_opensm_rhel_packages: []
eseries_ib_opensm_custom_packages:
  install:
    - files:
        add:
          "packages/opensm-libs-5.17.2.MLNX20240610.dc7c2998-
0.1.2310322.x86_64.rpm": "/tmp/"
          "packages/opensm-5.17.2.MLNX20240610.dc7c2998-
0.1.2310322.x86_64.rpm": "/tmp/"
    - packages:
        add:
          - /tmp/opensm-5.17.2.MLNX20240610.dc7c2998-
0.1.2310322.x86_64.rpm
          - /tmp/opensm-libs-5.17.2.MLNX20240610.dc7c2998-
0.1.2310322.x86_64.rpm
  uninstall:
    - packages:
        remove:
          - opensm
          - opensm-libs
      files:
        remove:
          - /tmp/opensm-5.17.2.MLNX20240610.dc7c2998-
0.1.2310322.x86_64.rpm
          - /tmp/opensm-libs-5.17.2.MLNX20240610.dc7c2998-
0.1.2310322.x86_64.rpm

eseries_ib_opensm_options:
  virt_enabled: "2"
```

Especifique la configuración de nodo de bloque común

Especifique la configuración de nodos de bloque común con las variables de grupo (group_var).

Descripción general

La configuración que debería Apple a todos los nodos de bloque se define en `group_vars/eseries_storage_systems.yml`. Normalmente incluye:

- Detalles sobre cómo el nodo de control Ansible debe conectarse a los sistemas de almacenamiento E-Series que se utilizan como nodos de bloques.

- Las versiones de firmware, NVSRAM y de unidad que deben ejecutar los nodos.
- Configuración global, que incluye la configuración de caché, la configuración de hosts y la configuración de cómo deben aprovisionarse los volúmenes.

> Las opciones establecidas en este archivo también pueden definirse en nodos de bloques individuales; por ejemplo, si se están utilizando modelos de hardware mixtos o tiene contraseñas diferentes para cada nodo. La configuración en nodos de bloque individuales tendrá prioridad sobre la configuración de este archivo.

Pasos

Cree el archivo `group_vars/eseries_storage_systems.yml` y rellenarlo de la siguiente manera:

1. Ansible no utiliza SSH para conectarse a los nodos de bloques y, en su lugar, utiliza API DE REST. Para lograrlo, debemos establecer:

```
ansible_connection: local
```

2. Especifique el nombre de usuario y la contraseña para gestionar cada nodo. El nombre de usuario puede omitirse opcionalmente (y, de forma predeterminada, admin), si no es posible especificar cualquier cuenta con privilegios de administrador. Especifique también si los certificados SSL deben verificarse o ignorarse:

```
eseries_system_username: admin
eseries_system_password: <PASSWORD>
eseries_validate_certs: false
```

> No se recomienda enumerar las contraseñas en texto sin formato. Use el almacén de Ansible o proporcione el `eseries_system_password` Al ejecutar Ansible con distribuidores de valor añadido de --extra.

3. Opcionalmente, especifique qué firmware de la controladora, NVSRAM y firmware de la unidad se debe instalar en los nodos. Deberá descargarse en el `packages/` directorio antes de ejecutar Ansible. El firmware de la controladora E-Series y NVSRAM se pueden descargar "aquí" y el firmware de la unidad "aquí":

```
eseries_firmware_firmware: "packages/<FILENAME>.dlp" # Ex.
"packages/RCB_11.80GA_6000_64cc0ee3.dlp"
eseries_firmware_nvsram: "packages/<FILENAME>.dlp" # Ex.
"packages/N6000-880834-D08.dlp"
eseries_drive_firmware_firmware_list:
  - "packages/<FILENAME>.dlp"
  # Additional firmware versions as needed.
eseries_drive_firmware_upgrade_drives_online: true # Recommended unless
BeeGFS hasn't been deployed yet, as it will disrupt host access if set
to "false".
```

Si se especifica esta configuración, Ansible actualizará automáticamente todo el firmware, incluido el reinicio de las controladoras (si es necesario) sin ningún aviso adicional. Se espera que esto no sea disruptivo para la I/o del host de BeeGFS, pero podría provocar un descenso temporal del rendimiento.

4. Ajuste los valores predeterminados de configuración global del sistema. BeeGFS en NetApp suele recomendar las opciones y valores que se incluyen en esta lista, pero se pueden ajustar en caso necesario:

```
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 by default.
```

5. Configure las opciones predeterminadas de aprovisionamiento de volúmenes globales. BeeGFS en NetApp suele recomendar las opciones y valores que se incluyen en esta lista, pero se pueden ajustar en caso necesario:

```
eseries_volume_size_unit: pct # Required by default. This allows volume
capacities to be specified as a percentage, simplifying putting together
the inventory.
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
```

6. Si es necesario, ajuste el orden en el que Ansible seleccionará las unidades para los pools de almacenamiento y los grupos de volúmenes, teniendo en cuenta las siguientes prácticas recomendadas:
   a. Enumere cualquier unidad (potencialmente menor) que se deben usar para los volúmenes de metadatos o gestión primero, y los volúmenes de almacenamiento en último lugar.
   b. Asegúrese de equilibrar el orden de selección de las unidades en los canales de unidad disponibles según los modelos de bandeja de discos/compartimento de unidades. Por ejemplo, con EF600 y sin expansiones, las unidades 0-11 están en el canal de unidades 1 y las unidades 12-23 están en el canal de unidades. Por lo tanto, una estrategia para equilibrar la selección de conducción es seleccionar `disk shelf:drive` 99:0, 99:23, 99:1, 99:22, etc. En el caso de que haya más de un compartimento, el primer dígito representa el ID de bandeja de unidades.

```
# Optimal/recommended order for the EF600 (no expansion):
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"
```

Haga clic en "aquí" para obtener un ejemplo de un archivo de inventario completo que representa la configuración común de nodos de bloques.

Defina los servicios BeeGFS

Defina el servicio de gestión de BeeGFS

Los servicios BeeGFS se configuran mediante variables de grupo (Group_var).

Descripción general

En esta sección se describe la definición del servicio de gestión de BeeGFS. En los clústeres de alta disponibilidad solo debe haber un servicio de este tipo para un sistema de archivos concreto. La configuración de este servicio incluye la definición:

- El tipo de servicio (gestión).
- Definir cualquier configuración que sólo se debe aplicar a este servicio BeeGFS.
- Configuración de una o varias IP flotantes (interfaces lógicas) en las que se puede acceder a este servicio.
- Especificar dónde y cómo debe almacenar un volumen datos para este servicio (el objetivo de gestión de BeeGFS).

Pasos

Cree un nuevo archivo `group_vars/mgmt.yml` y haga referencia a la "Planifique el sistema de archivos" sección. Llévelo de la siguiente manera:

1. Indique que este archivo representa la configuración de un servicio de administración de BeeGFS:

```
beegfs_service: management
```

2. Defina cualquier configuración que se deba aplicar sólo a este servicio BeeGFS. Esto no suele ser necesario para el servicio de gestión a menos que necesite habilitar cuotas, sin embargo, con cualquier parámetro de configuración admitido de `beegfs-mgmtd.conf` se puede incluir. Nota los siguientes parámetros se configuran automáticamente u otros lugares y no se deben especificar aquí: `storeMgmtdDirectory`, `connAuthFile`, `connDisableAuthentication`, `connInterfacesFile`, y. `connNetFilterFile`.

```
beegfs_ha_beegfs_mgmtd_conf_resource_group_options:
  <beegfs-mgmt.conf:key>:<beegfs-mgmt.conf:value>
```

3. Configure uno o varios IP flotantes que utilizarán otros servicios y clientes para conectarse a este servicio (esto establecerá automáticamente BeeGFS `connInterfacesFile` opción):

```
floating_ips:
  - <INTERFACE>:<IP/SUBNET> # Primary interface. Ex.
i1b:100.127.101.0/16
  - <INTERFACE>:<IP/SUBNET> # Secondary interface(s) as needed.
```

4. Opcionalmente, especifique una o varias subredes IP permitidas que se pueden utilizar para la comunicación saliente (esto establecerá automáticamente BeeGFS `connNetFilterFile` opción):

```
filter_ip_ranges:
  - <SUBNET>/<MASK> # Ex. 192.168.10.0/24
```

5. Especifique el objetivo de gestión de BeeGFS en el que este servicio almacenará datos de acuerdo con las siguientes directrices:
   a. Se puede utilizar el mismo nombre de pool de almacenamiento o grupo de volúmenes para varios servicios/objetivos de BeeGFS; asegúrese de utilizar el mismo `name`, `raid_level`, `criteria_*`, y. `common_*` la configuración de cada uno (los volúmenes enumerados para cada servicio deben ser diferentes).
   b. Los tamaños de los volúmenes se deben especificar como un porcentaje del pool de almacenamiento/grupo de volúmenes y el total no debe ser superior a 100 en todos los servicios/volúmenes que utilizan un pool de almacenamiento/grupo de volúmenes en particular. Nota Cuando se usan SSD, se recomienda dejar un poco de espacio libre en el grupo de volúmenes para maximizar el rendimiento de SSD y la vida útil (haga clic "aquí" para obtener más detalles).
   c. Haga clic en "aquí" para obtener una lista completa de las opciones de configuración disponibles para `eseries_storage_pool_configuration`. Tenga en cuenta algunas opciones como `state`, `host`, `host_type`, `workload_name`, y. `workload_metadata` y los nombres de volúmenes se generan automáticamente y no se deben especificar aquí.

```
beegfs_targets:
  <BLOCK_NODE>: # The name of the block node as found in the Ansible
inventory. Ex: netapp_01
    eseries_storage_pool_configuration:
      - name: <NAME> # Ex: beegfs_m1_m2_m5_m6
        raid_level: <LEVEL> # One of: raid1, raid5, raid6, raidDiskPool
        criteria_drive_count: <DRIVE COUNT> # Ex. 4
        common_volume_configuration:
          segment_size_kb: <SEGMENT SIZE> # Ex. 128
        volumes:
          - size: <PERCENT> # Percent of the pool or volume group to
allocate to this volume. Ex. 1
            owning_controller: <CONTROLLER> # One of: A, B
```

Haga clic en "aquí" Para obtener un ejemplo de un archivo de inventario completo que representa un servicio de administración de BeeGFS.

Defina el servicio de metadatos BeeGFS

Los servicios BeeGFS se configuran mediante variables de grupo (Group_var).

Descripción general

En esta sección se describe la definición del servicio de metadatos de BeeGFS. Al menos debe haber un servicio de este tipo en los clústeres de alta disponibilidad para un sistema de archivos determinado. La configuración de este servicio incluye la definición:

- El tipo de servicio (metadatos).
- Definir cualquier configuración que sólo se debe aplicar a este servicio BeeGFS.
- Configuración de una o varias IP flotantes (interfaces lógicas) en las que se puede acceder a este servicio.
- Especificar dónde y cómo debe almacenar un volumen datos para este servicio (el objetivo de metadatos BeeGFS).

Pasos

Haciendo referencia a la "Planifique el sistema de archivos" sección, cree un archivo en `group_vars/meta_<ID>.yml` para cada servicio de metadatos del cluster y rellénelo de la siguiente manera:

1. Indique que este archivo representa la configuración de un servicio de metadatos BeeGFS:

```
beegfs_service: metadata
```

2. Defina cualquier configuración que se deba aplicar sólo a este servicio BeeGFS. Al mínimo debe especificar el puerto TCP y UDP deseado, sin embargo, cualquier parámetro de configuración compatible desde `beegfs-meta.conf` también se puede incluir. Nota los siguientes parámetros se configuran automáticamente u otros lugares y no se deben especificar aquí: `sysMgmtdHost`, `storeMetaDirectory`, `connAuthFile`, `connDisableAuthentication`, `connInterfacesFile`, y. `connNetFilterFile`.

```
beegfs_ha_beegfs_meta_conf_resource_group_options:
  connMetaPortTCP: <TCP PORT>
  connMetaPortUDP: <UDP PORT>
  tuneBindToNumaZone: <NUMA ZONE> # Recommended if using file nodes with
multiple CPU sockets.
```

3. Configure uno o varios IP flotantes que utilizarán otros servicios y clientes para conectarse a este servicio (esto establecerá automáticamente BeeGFS `connInterfacesFile` opción):

```
floating_ips:
  - <INTERFACE>:<IP/SUBNET> # Primary interface. Ex.
i1b:100.127.101.1/16
  - <INTERFACE>:<IP/SUBNET> # Secondary interface(s) as needed.
```

4. Opcionalmente, especifique una o varias subredes IP permitidas que se pueden utilizar para la comunicación saliente (esto establecerá automáticamente BeeGFS `connNetFilterFile` opción):

```
filter_ip_ranges:
  - <SUBNET>/<MASK> # Ex. 192.168.10.0/24
```

5. Especifique el destino de metadatos BeeGFS en el que este servicio almacenará datos de acuerdo con las siguientes directrices (también configurará automáticamente el `storeMetaDirectory` opción):
   a. Se puede utilizar el mismo nombre de pool de almacenamiento o grupo de volúmenes para varios servicios/objetivos de BeeGFS; asegúrese de utilizar el mismo `name`, `raid_level`, `criteria_*`, y. `common_*` la configuración de cada uno (los volúmenes enumerados para cada servicio deben ser diferentes).
   b. Los tamaños de los volúmenes se deben especificar como un porcentaje del pool de almacenamiento/grupo de volúmenes y el total no debe ser superior a 100 en todos los servicios/volúmenes que utilizan un pool de almacenamiento/grupo de volúmenes en particular. Nota Cuando se usan SSD, se recomienda dejar un poco de espacio libre en el grupo de volúmenes para maximizar el rendimiento de SSD y la vida útil (haga clic "aquí" para obtener más detalles).
   c. Haga clic en "aquí" para obtener una lista completa de las opciones de configuración disponibles para `eseries_storage_pool_configuration`. Tenga en cuenta algunas opciones como `state`, `host`, `host_type`, `workload_name`, y. `workload_metadata` y los nombres de volúmenes se generan automáticamente y no se deben especificar aquí.

```
beegfs_targets:
  <BLOCK_NODE>: # The name of the block node as found in the Ansible
inventory. Ex: netapp_01
    eseries_storage_pool_configuration:
      - name: <NAME> # Ex: beegfs_m1_m2_m5_m6
        raid_level: <LEVEL> # One of: raid1, raid5, raid6, raidDiskPool
        criteria_drive_count: <DRIVE COUNT> # Ex. 4
        common_volume_configuration:
          segment_size_kb: <SEGMENT SIZE> # Ex. 128
        volumes:
          - size: <PERCENT> # Percent of the pool or volume group to
allocate to this volume. Ex. 1
            owning_controller: <CONTROLLER> # One of: A, B
```

Haga clic en "aquí" Para obtener un ejemplo de un archivo de inventario completo que representa un servicio de metadatos BeeGFS.

Defina el servicio de almacenamiento BeeGFS

Los servicios BeeGFS se configuran mediante variables de grupo (Group_var).

Descripción general

En esta sección se describe la definición del servicio de almacenamiento de BeeGFS. Al menos debe haber un servicio de este tipo en los clústeres de alta disponibilidad para un sistema de archivos determinado. La

configuración de este servicio incluye la definición:

- El tipo de servicio (almacenamiento).
- Definir cualquier configuración que sólo se debe aplicar a este servicio BeeGFS.
- Configuración de una o varias IP flotantes (interfaces lógicas) en las que se puede acceder a este servicio.
- Especificar dónde/cómo deben ser los volúmenes para almacenar datos para este servicio (los destinos de almacenamiento BeeGFS).

Pasos

Haciendo referencia a la "Planifique el sistema de archivos" sección, cree un archivo en `group_vars/stor_<ID>.yml` para cada servicio de almacenamiento del clúster y rellénelo de la siguiente manera:

1. Indique este archivo que representa la configuración de un servicio de almacenamiento BeeGFS:

```
beegfs_service: storage
```

2. Defina cualquier configuración que se deba aplicar sólo a este servicio BeeGFS. Al mínimo debe especificar el puerto TCP y UDP deseado, sin embargo, cualquier parámetro de configuración compatible desde `beegfs-storage.conf` también se puede incluir. Nota los siguientes parámetros se configuran automáticamente u otros lugares y no se deben especificar aquí: `sysMgmtdHost`, `storeStorageDirectory`, `connAuthFile`, `connDisableAuthentication`, `connInterfacesFile`, y. `connNetFilterFile`.

```
beegfs_ha_beegfs_storage_conf_resource_group_options:
  connStoragePortTCP: <TCP PORT>
  connStoragePortUDP: <UDP PORT>
  tuneBindToNumaZone: <NUMA ZONE> # Recommended if using file nodes with
multiple CPU sockets.
```

3. Configure uno o varios IP flotantes que utilizarán otros servicios y clientes para conectarse a este servicio (esto establecerá automáticamente BeeGFS `connInterfacesFile` opción):

```
floating_ips:
  - <INTERFACE>:<IP/SUBNET> # Primary interface. Ex.
i1b:100.127.101.1/16
  - <INTERFACE>:<IP/SUBNET> # Secondary interface(s) as needed.
```

4. Opcionalmente, especifique una o varias subredes IP permitidas que se pueden utilizar para la comunicación saliente (esto establecerá automáticamente BeeGFS `connNetFilterFile` opción):

```
filter_ip_ranges:
  - <SUBNET>/<MASK> # Ex. 192.168.10.0/24
```

5. Especifique los objetivos de almacenamiento de BeeGFS en los que este servicio almacenará datos de acuerdo con las siguientes directrices (también configurará automáticamente el `storeStorageDirectory` opción):
   a. Se puede utilizar el mismo nombre de pool de almacenamiento o grupo de volúmenes para varios servicios/objetivos de BeeGFS; asegúrese de utilizar el mismo `name`, `raid_level`, `criteria_*`, y. `common_*` la configuración de cada uno (los volúmenes enumerados para cada servicio deben ser diferentes).
   b. Los tamaños de los volúmenes se deben especificar como un porcentaje del pool de almacenamiento/grupo de volúmenes y el total no debe ser superior a 100 en todos los servicios/volúmenes que utilizan un pool de almacenamiento/grupo de volúmenes en particular. Nota Cuando se usan SSD, se recomienda dejar un poco de espacio libre en el grupo de volúmenes para maximizar el rendimiento de SSD y la vida útil (haga clic "aquí" para obtener más detalles).
   c. Haga clic en "aquí" para obtener una lista completa de las opciones de configuración disponibles para `eseries_storage_pool_configuration`. Tenga en cuenta algunas opciones como `state`, `host`, `host_type`, `workload_name`, y. `workload_metadata` y los nombres de volúmenes se generan automáticamente y no se deben especificar aquí.

```
beegfs_targets:
  <BLOCK_NODE>: # The name of the block node as found in the Ansible
inventory. Ex: netapp_01
    eseries_storage_pool_configuration:
      - name: <NAME> # Ex: beegfs_s1_s2
        raid_level: <LEVEL> # One of: raid1, raid5, raid6,
raidDiskPool
        criteria_drive_count: <DRIVE COUNT> # Ex. 4
        common_volume_configuration:
          segment_size_kb: <SEGMENT SIZE> # Ex. 128
        volumes:
          - size: <PERCENT> # Percent of the pool or volume group to
allocate to this volume. Ex. 1
            owning_controller: <CONTROLLER> # One of: A, B
        # Multiple storage targets are supported / typical:
          - size: <PERCENT> # Percent of the pool or volume group to
allocate to this volume. Ex. 1
            owning_controller: <CONTROLLER> # One of: A, B
```

Haga clic en "aquí" Para obtener un ejemplo de un archivo de inventario completo que representa un servicio de almacenamiento BeeGFS.

Asigne servicios BeeGFS a los nodos de archivo

Especifique qué nodos de archivo pueden ejecutar cada servicio BeeGFS mediante `inventory.yml` archivo.

Descripción general

En esta sección se explica cómo crear la `inventory.yml` archivo. Esto incluye una lista de todos los nodos de bloque y especificar qué nodos de archivo pueden ejecutar cada servicio BeeGFS.

Pasos

Cree el archivo `inventory.yml` y rellenarlo de la siguiente manera:

1. Desde la parte superior del archivo, cree la estructura de inventario estándar de Ansible:

```
# BeeGFS HA (High_Availability) cluster inventory.
all:
  children:
```

2. Cree un grupo que contenga todos los nodos de bloques que participan en este clúster de alta disponibilidad:

```
    # Ansible group representing all block nodes:
    eseries_storage_systems:
      hosts:
        <BLOCK NODE HOSTNAME>:
        <BLOCK NODE HOSTNAME>:
        # Additional block nodes as needed.
```

3. Cree un grupo que contendrá todos los servicios BeeGFS del clúster y los nodos de archivo que los ejecutarán:

```
    # Ansible group representing all file nodes:
    ha_cluster:
      children:
```

4. Para cada servicio BeeGFS del clúster, defina los nodos de archivos preferidos y secundarios que deben ejecutar ese servicio:

```
        <SERVICE>: # Ex. "mgmt", "meta_01", or "stor_01".
          hosts:
            <FILE NODE HOSTNAME>:
            <FILE NODE HOSTNAME>:
            # Additional file nodes as needed.
```

Haga clic en "aquí" para obtener un ejemplo de un archivo de inventario completo.

Implemente el sistema de archivos BeeGFS

Descripción general del libro de estrategia de Ansible

Puesta en marcha y gestión de clústeres de alta disponibilidad de BeeGFS mediante Ansible.

Descripción general

En las secciones anteriores se han realizado los pasos necesarios para crear un inventario de Ansible que represente un clúster de alta disponibilidad de BeeGFS. En esta sección se presenta la automatización de Ansible desarrollada por NetApp para poner en marcha y gestionar el clúster.

Ansible: Conceptos clave

Antes de continuar, es útil estar familiarizado con algunos conceptos clave de Ansible:

- Las tareas que se deben ejecutar con un inventario de Ansible se definen en lo que se conoce como **playbook**.
  - La mayoría de las tareas en Ansible están diseñadas para ser **idempotente**, lo que significa que pueden ejecutarse varias veces para verificar que la configuración/estado deseada todavía se aplica sin romper las cosas ni hacer actualizaciones innecesarias.
- La unidad más pequeña de ejecución en Ansible es un **módulo**.
  - Los libros de estrategia habituales utilizan varios módulos.
    - Ejemplos: Descargue un paquete, actualice un archivo de configuración, inicie/habilite un servicio.
  - NetApp distribuye módulos para automatizar los sistemas E-Series de NetApp.
- La automatización compleja se empaquetará mejor como rol.
  - Básicamente, un formato estándar para distribuir un libro de aplicaciones reutilizable.
  - NetApp distribuye roles para hosts Linux y sistemas de archivos BeeGFS.

Rol de ha de BeeGFS para Ansible: Conceptos clave

Toda la automatización necesaria para poner en marcha y gestionar cada versión de BeeGFS en NetApp se presenta como un rol de Ansible y se distribuye como parte de la "Colección de Ansible E-Series de NetApp para BeeGFS":

- Este papel se puede considerar como un lugar entre un motor de **instalador** y un motor de **implementación/administración** moderno para BeeGFS.
  - Aplica una infraestructura moderna como prácticas de código y filosofías para simplificar la gestión de infraestructura de almacenamiento a cualquier escala.
  - De forma similar a cómo el "Kubespray" proyecto permite a los usuarios implementar y mantener toda una distribución de Kubernetes para una infraestructura informática de escalado horizontal.
- Este rol es el formato **definido por software** que utiliza NetApp para empaquetar, distribuir y mantener BeeGFS en soluciones NetApp.
  - Esforzarse por crear una experiencia “similar a un dispositivo” sin necesidad de distribuir una distribución entera de Linux o una imagen grande.
  - Incluye agentes de recursos en clúster compatibles con Open Cluster Framework (OCF) de NetApp

para objetivos BeeGFS personalizados, direcciones IP y supervisión que proporcionan una integración inteligente con Pacemaker/BeeGFS.

- Esta función no se limita a la puesta en marcha de la «automatización», sino que está destinada a gestionar todo el ciclo de vida del sistema de archivos, incluidos:
  - Aplicar cambios y actualizaciones de configuración por servicio o para todo el clúster.
  - Automatizar la reparación y recuperación del clúster después de resolver problemas de hardware.
  - Simplificación del ajuste del rendimiento con valores predeterminados establecidos en función de amplias pruebas con BeeGFS y volúmenes de NetApp.
  - Verificación y corrección de deriva de configuración.

NetApp también proporciona un rol de Ansible para "Clientes de BeeGFS", Que se puede utilizar opcionalmente para instalar BeeGFS y montar sistemas de archivos en nodos de cálculo/GPU/inicio de sesión.

Ponga en marcha el clúster de alta disponibilidad de BeeGFS

Especifique qué tareas se deben ejecutar para poner en marcha el clúster de alta disponibilidad de BeeGFS mediante un libro de aplicaciones.

Descripción general

En esta sección se describe cómo montar el libro de estrategia estándar utilizado para poner en marcha/gestionar BeeGFS en NetApp.

Pasos

Cree el libro de aplicaciones de Ansible

Cree el archivo `playbook.yml` y rellenarlo de la siguiente manera:

1. Defina primero un conjunto de tareas (comúnmente denominado a) "repr") Que solo se debe ejecutar en los nodos de bloques E-Series de NetApp. Utilizamos una tarea de pausa para preguntar antes de ejecutar la instalación (para evitar la ejecución accidental de playbook) y, a continuación, importar la `nar_santricity_management` función. Esta función se ocupa de aplicar cualquier configuración general del sistema definida en `group_vars/eseries_storage_systems.yml` o individual `host_vars/<BLOCK NODE>.yml` archivos.

```
- hosts: eseries_storage_systems
  gather_facts: false
  collections:
    - netapp_eseries.santricity
  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: Configure NetApp E-Series block nodes.
      import_role:
        name: nar_santricity_management
```

2. Defina la reproducción que se ejecutará en todos los nodos de archivos y bloques:

```
- hosts: all
  any_errors_fatal: true
  gather_facts: false
  collections:
    - netapp_eseries.beegfs
```

3. En esta aplicación podemos definir opcionalmente un conjunto de «tareas previas» que se deben ejecutar antes de poner en marcha el clúster de alta disponibilidad. Esto puede ser útil para verificar/instalar requisitos previos como Python. También podemos inyectar comprobaciones previas al vuelo, por ejemplo, verificar que las etiquetas de Ansible proporcionadas son compatibles:

```
  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 }}"
```

4. Por último, este juego importa el rol de ha de BeeGFS para la versión de BeeGFS que desea implementar:

```
  tasks:
    - name: Verify the BeeGFS HA cluster is properly deployed.
      import_role:
        name: beegfs_ha_7_4 # Alternatively specify: beegfs_ha_7_3.
```

> Se mantiene un rol de ha de BeeGFS para cada versión principal.secundaria de BeeGFS admitida. Esto permite a los usuarios elegir cuándo desean actualizar versiones principales o secundarias. Actualmente (`beegfs_7_3`(`beegfs_7_2`se admiten BeeGFS 7,3.x) o BeeGFS 7,2.x). De forma predeterminada, ambos roles implementarán la versión más reciente del parche de BeeGFS en el momento de su publicación, aunque los usuarios pueden optar por anular este parche e implementar el último parche si lo desean. Consulte la última "guía de actualización"información para obtener más información.

5. Opcional: Si desea definir tareas adicionales, tenga en cuenta si las tareas deben ser dirigidas `all` Los hosts (incluidos los sistemas de almacenamiento E-Series) o solo los nodos de archivos. Si es necesario, defina una nueva reproducción dirigida específicamente a los nodos de archivo mediante `- hosts: ha_cluster`.

Haga clic en "aquí" por ejemplo, un archivo de libro de estrategia completo.

Instale las colecciones Ansible de NetApp

Se mantiene la colección BeeGFS para Ansible y todas las dependencias "Galaxia de ansible". En su nodo de control de Ansible, ejecute el siguiente comando para instalar la versión más reciente:

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

Aunque normalmente no se recomienda, también es posible instalar una versión específica de la colección:

```
ansible-galaxy collection install netapp_eseries.beegfs:
==<MAJOR>.<MINOR>.<PATCH>
```

Ejecute el libro de aplicaciones

Desde el directorio del nodo de control de Ansible que contiene el `inventory.yml` y.. `playbook.yml` ejecute el libro de estrategia de la siguiente forma:

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

Según el tamaño del clúster, la puesta en marcha inicial puede tardar más de 20 minutos. Si se produce un error en la puesta en marcha por algún motivo, solo tiene que corregir los problemas (p. ej., un cableado incorrecto, nodo no se inició, etc.) y reiniciar el libro de estrategia de Ansible.

Al especificar "configuración común de nodos de archivos", si selecciona la opción predeterminada para que Ansible gestione automáticamente la autenticación basada en conexión, `connAuthFile` ahora se puede encontrar un secreto utilizado como secreto compartido en `<playbook_dir>/files/beegfs/<sysMgmtdHost>_connAuthFile` (de forma predeterminada). Cualquier cliente que necesite acceder al sistema de archivos tendrá que utilizar este secreto compartido. Esto se gestiona automáticamente si los clientes se configuran mediante el "Función de cliente de BeeGFS".

Implemente clientes BeeGFS

Opcionalmente, Ansible se puede usar para configurar los clientes de BeeGFS y montar

el sistema de archivos.

Descripción general

Para acceder a los sistemas de archivos BeeGFS es necesario instalar y configurar el cliente BeeGFS en cada nodo que necesite montar el sistema de archivos. En esta sección se documenta cómo realizar estas tareas mediante el útil disponible "Rol de Ansible".

Pasos

Cree el archivo de inventario de cliente

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>
```

3. Incluya de manera opcional uno de los siguientes elementos si desea utilizar los roles de recogida de hosts E-Series de NetApp para configurar las interfaces InfiniBand o Ethernet para que los clientes se conecten a los nodos de archivos BeeGFS:
   a. Si el tipo de red es "InfiniBand (uso de IPoIB)":

```
eseries_ipoib_interfaces:
- name: <INTERFACE>  # Example: ib0 or i1b
  address: <IP/SUBNET> # Example: 100.127.100.1/16
- name: <INTERFACE>  # Additional interfaces as needed.
  address: <IP/SUBNET>
```

   b. Si el tipo de red es "RDMA sobre Ethernet convergente (roce)":

```
eseries_roce_interfaces:
- name: <INTERFACE>  # Example: eth0.
  address: <IP/SUBNET> # Example: 100.127.100.1/16
- name: <INTERFACE>  # Additional interfaces as needed.
  address: <IP/SUBNET>
```

   c. Si el tipo de red es "Ethernet (solo TCP, sin RDMA)":

```
eseries_ip_interfaces:
- name: <INTERFACE>  # Example: eth0.
  address: <IP/SUBNET> # Example: 100.127.100.1/16
- name: <INTERFACE>  # Additional interfaces as needed.
  address: <IP/SUBNET>
```

4. Cree un archivo nuevo `client_inventory.yml` Y especifique el usuario que Ansible debe usar para conectarse a cada cliente y la contraseña que Ansible debe usar para el escalado de privilegios (esto requiere `ansible_ssh_user` sea raíz o tenga privilegios sudo):

```
# BeeGFS client inventory.
all:
  vars:
    ansible_ssh_user: <USER>
    ansible_become_password: <PASSWORD>
```

> No almacene contraseñas en texto sin formato. En su lugar, utilice Ansible Vault (consulte "Documentación de Ansible" Para cifrar contenido con Ansible Vault) o usar el `--ask -become-pass` al ejecutar el libro de estrategia.

5. En la `client_inventory.yml` File, enumera todos los hosts que deben configurarse como clientes BeeGFS en `beegfs_clients` Agrupe y, a continuación, consulte los comentarios en línea y elimine los comentarios de cualquier configuración adicional necesaria para crear el módulo de kernel de cliente BeeGFS en su sistema:

```
children:
    # Ansible group representing all BeeGFS clients:
    beegfs_clients:
      hosts:
        <CLIENT HOSTNAME>:
        # Additional clients as needed.

      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 de acceso de inclusión de encabezado correcta para la instalación de Linux. Para obtener más información, consulte la documentación de BeeGFS para "Compatibilidad con RDMA".

6. En la `client_inventory.yml` Archivo, enumere los sistemas de archivos BeeGFS que desea montar en cualquiera de los definidos previamente `vars`:

```
        beegfs_client_mounts:
          - sysMgmtdHost: <IP ADDRESS>  # 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
        # Specify additional file system mounts for this or other file
systems.
```

7. A partir de BeeGFS 7.2.7 y 7.3.1 "autenticación de conexión"se deben configurar o desactivar explícitamente. Dependiendo de cómo decida configurar la autenticación basada en conexión al especificar "configuración común de nodos de archivos", puede que necesite ajustar la configuración del cliente:
   a. De forma predeterminada, la puesta en marcha del clúster de alta disponibilidad configurará automáticamente la autenticación de conexiones y generará un `connauthfile` Que se colocará/mantendrá en el nodo de control de Ansible en `<INVENTORY>/files/beegfs/<sysMgmtdHost>_connAuthFile`. De forma predeterminada, la función de cliente BeeGFS está configurada para leer/distribuir este archivo a los clientes definidos en `client_inventory.yml`, y no se necesita ninguna acción adicional.
      i. Para obtener información sobre las opciones avanzadas, consulte la lista completa de valores predeterminados que se incluyen con la "Función de cliente de BeeGFS".
   b. Si decide especificar un secreto personalizado con `beegfs_ha_conn_auth_secret` especifique en la `client_inventory.yml` también archivo:

      ```
      beegfs_ha_conn_auth_secret: <SECRET>
      ```

   c. Si decide deshabilitar la autenticación basada en conexión completamente con `beegfs_ha_conn_auth_enabled`, especifique que en la `client_inventory.yml` también

archivo:

```
beegfs_ha_conn_auth_enabled: false
```

Para obtener una lista completa de los parámetros admitidos y detalles adicionales, consulte la "Documentación completa del cliente de BeeGFS". Para ver un ejemplo completo de un inventario de cliente, haga clic en "aquí".

Cree el archivo del libro de aplicaciones del cliente BeeGFS

1. Cree un archivo nuevo `client_playbook.yml`

```
# BeeGFS client playbook.
- hosts: beegfs_clients
  any_errors_fatal: true
  gather_facts: true
  collections:
    - netapp_eseries.beegfs
    - netapp_eseries.host
  tasks:
```

2. Opcional: Si desea utilizar los roles de la recogida de hosts de E-Series de NetApp para configurar interfaces para que los clientes se conecten a sistemas de archivos BeeGFS, importe el rol correspondiente al tipo de interfaz que está configurando:
   a. Si utiliza InfiniBand (IPoIB):

```
    - name: Ensure IPoIB is configured
      import_role:
        name: ipoib
```

   b. Si utiliza RDMA over Converged Ethernet (roce):

```
    - name: Ensure IPoIB is configured
      import_role:
        name: roce
```

   c. Si utiliza Ethernet (solo TCP, no RDMA):

```
    - name: Ensure IPoIB is configured
      import_role:
        name: ip
```

3. Por último, importe la función de cliente de BeeGFS para instalar el software cliente y configurar los montajes del sistema de archivos:

```
    # REQUIRED: Install the BeeGFS client and mount the BeeGFS file
system.
    - name: Verify the BeeGFS clients are configured.
      import_role:
        name: beegfs_client
```

Para ver un ejemplo completo de un libro de aplicaciones del cliente, haga clic en "aquí".

Ejecute el libro de aplicaciones del cliente BeeGFS

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

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

Verifique la implementación de BeeGFS

Compruebe la implementación del sistema de archivos antes de colocar el sistema en producción.

Descripción general

Antes de poner el sistema de archivos BeeGFS en producción, realice algunas comprobaciones de verificación.

Pasos

1. Inicie sesión en cualquier cliente y ejecute lo siguiente para garantizar que todos los nodos esperados estén presentes o sean accesibles, no se han notificado inconsistencias ni otros problemas:

```
beegfs-fsck --checkfs
```

2. Apague todo el clúster y, a continuación, reinícielo. Desde cualquier nodo de archivo ejecute lo siguiente:

```
pcs cluster stop --all # Stop the cluster on all file nodes.
pcs cluster start --all  # Start the cluster on all file nodes.
pcs status # Verify all nodes and services are started and no failures
are reported (the command may need to be reran a few times to allow time
for all services to start).
```

3. Coloque cada nodo en espera y compruebe que los servicios de BeeGFS pueden conmutar por error a nodos secundarios. Para realizar este inicio de sesión en cualquiera de los nodos de archivo y ejecute lo siguiente:

```
pcs status # Verify the cluster is healthy at the start.
pcs node standby <FILE NODE HOSTNAME> # Place the node under test in
standby.
pcs status # Verify services are started on a secondary node and no
failures are reported.
pcs node unstandby <FILE NODE HOSTNAME> # Take the node under test out
of standby.
pcs status # Verify the file node is back online and no failures are
reported.
pcs resource relocate run # Move all services back to their preferred
nodes.
pcs status # Verify services have moved back to the preferred node.
```

4. Utilizar herramientas de evaluación del rendimiento como IOR y MDTest para verificar que el rendimiento del sistema de archivos cumple las expectativas. Se pueden encontrar ejemplos de pruebas y parámetros comunes utilizados con BeeGFS "Verificación del diseño"en la sección de BeeGFS on NetApp Verified Architecture.

Las pruebas adicionales deben realizarse en función de los criterios de aceptación definidos para una instalación/emplazamiento particular.

Administre clústeres BeeGFS

Descripción general, conceptos clave y terminología

Obtenga información sobre cómo administrar los clústeres de alta disponibilidad de BeeGFS después de haberse puesto en marcha.

Descripción general

Esta sección está destinada a administradores de clústeres que deben gestionar clústeres de alta disponibilidad de BeeGFS una vez que se han puesto en marcha. Incluso los que estén familiarizados con los clústeres de alta disponibilidad de Linux deben leer esta guía detenidamente, ya que hay varias diferencias en cómo gestionar el clúster, especialmente en lo que respecta a la reconfiguración debido al uso de Ansible.

Conceptos clave

Aunque algunos de estos conceptos se introducen en "términos y conceptos"la página principal, es útil reintroducirlos en el contexto de un clúster de alta disponibilidad de BeeGFS:

Nodo de clúster: servidor que ejecuta los servicios Pacemaker y Corosync y que participa en el clúster ha.

Nodo de archivo: un nodo de clúster utilizado para ejecutar uno o más servicios de gestión, metadatos o almacenamiento de BeeGFS.

Nodo de bloque: Un sistema de almacenamiento E-Series de NetApp que proporciona almacenamiento basado en bloques a los nodos de ficheros. Estos nodos no participan en el clúster de alta disponibilidad de BeeGFS porque proporcionan sus propias funcionalidades de alta disponibilidad independientes. Cada nodo consta de dos controladoras de almacenamiento que proporcionan alta disponibilidad en la capa de bloques.

Servicio BeeGFS: un servicio de gestión, metadatos o almacenamiento de BeeGFS. Cada nodo de archivo ejecutará uno o más servicios que usarán volúmenes en el nodo de bloque para almacenar sus datos.

Elementos básicos: una puesta en marcha estandarizada de nodos de archivos BeeGFS, nodos de bloque E-Series y servicios BeeGFS que se ejecutan en ellos simplifican el escalado de un clúster/sistema de archivos de alta disponibilidad de BeeGFS siguiendo una arquitectura verificada de NetApp. También se admiten clústeres de alta disponibilidad personalizados, pero a menudo siguen un método de elementos básicos similar para simplificar el escalado.

BeeGFS ha Cluster: un número escalable de nodos de archivo utilizados para ejecutar servicios BeeGFS respaldados por nodos de bloque para almacenar los datos de BeeGFS de forma muy disponible. Desarrollado a partir de componentes de código abierto demostrados en el sector Pacemaker y Corosync con Ansible para el paquete y la puesta en marcha.

Servicios de Cluster Server: se refiere a los servicios Pacemaker y Corosync que se ejecutan en cada nodo que participa en el cluster. Tenga en cuenta que es posible que un nodo no ejecute ningún servicio BeeGFS y solo participe en el clúster como nodo "tiebreaker" en el caso de que solo haya necesidad de dos nodos de archivo.

Recursos de clúster: para cada servicio BeeGFS que se ejecuta en el clúster, verá un recurso de supervisión BeeGFS y un grupo de recursos que contiene recursos para los objetivos BeeGFS, direcciones IP (IP flotantes) y el servicio BeeGFS mismo.

Ansible: una herramienta para el aprovisionamiento de software, la gestión de la configuración y la puesta en

marcha de aplicaciones, lo que permite la infraestructura como código. Así es como los clústeres de alta disponibilidad de BeeGFS se empaquetan para simplificar el proceso de puesta en marcha, reconfiguración y actualización de BeeGFS en NetApp.

pc: una interfaz de línea de comandos disponible desde cualquiera de los nodos de archivos del clúster utilizados para consultar y controlar el estado de los nodos y recursos del clúster.

Terminología común

Failover: cada servicio BeeGFS tiene un nodo de archivo preferido en el que se ejecutará a menos que dicho nodo falle. Cuando un servicio BeeGFS se ejecuta en un nodo de archivo no preferido/secundario, se dice que está en conmutación por error.

Failback: el acto de mover servicios BeeGFS de un nodo de archivo no preferido de nuevo a su nodo preferido.

Par de alta disponibilidad: dos nodos de archivo que pueden acceder al mismo conjunto de nodos de bloque se denominan a veces pares de alta disponibilidad. Este es un término común que se emplea en NetApp para hacer referencia a dos controladoras de almacenamiento o nodos que se pueden «sustituir» entre sí.

Modo de mantenimiento: desactiva la supervisión de todos los recursos e impide que Pacemaker mueva o administre de otro modo los recursos en el clúster (consulte también la sección en "modo de mantenimiento").

Clúster de alta disponibilidad: uno o más nodos de archivo que ejecutan servicios BeeGFS que pueden conmutar por error entre varios nodos del clúster para crear un sistema de archivos BeeGFS de alta disponibilidad. A menudo, los nodos de archivo se configuran en pares de alta disponibilidad que pueden ejecutar un subconjunto de los servicios de BeeGFS del clúster.

Cuándo usar Ansible frente a la herramienta pc

¿Cuándo debe utilizar Ansible frente a la herramienta de línea de comandos de pc para gestionar el clúster de alta disponibilidad?

Todas las tareas de puesta en marcha y reconfiguración del clúster se deben completar con Ansible desde un nodo de control externo de Ansible. Normalmente, los cambios temporales en el estado del clúster (por ejemplo, la colocación de nodos dentro y fuera de espera) se realizarán iniciando sesión en un nodo del clúster (preferiblemente uno que no esté degradado o esté a punto de someterse a tareas de mantenimiento) y utilizando la herramienta de línea de comandos del pc.

La modificación de cualquier configuración del clúster, incluidos los recursos, las restricciones, las propiedades y los propios servicios de BeeGFS, siempre debe realizarse con Ansible. Mantener una copia actualizada del inventario y el libro de estrategia de Ansible (lo ideal para un control de origen para realizar un seguimiento de los cambios) forma parte del mantenimiento del clúster. Si necesita realizar cambios en la configuración, actualice el inventario y vuelva a ejecutar el libro de aplicaciones de Ansible que importa el rol de alta disponibilidad de BeeGFS.

El rol de alta disponibilidad manejará colocar el clúster en modo de mantenimiento y, a continuación, hará los cambios necesarios antes de reiniciar BeeGFS o los servicios de clúster para aplicar la nueva configuración. Dado que normalmente no es necesario reiniciar el nodo completo fuera de la puesta en marcha inicial, Ansible se considera un procedimiento "seguro", pero siempre es un procedimiento recomendado durante las ventanas de mantenimiento o fuera de las horas, en caso de que se deban reiniciar los servicios de BeeGFS. Normalmente, estos reinicios no deben causar errores en la aplicación, pero pueden afectar al rendimiento (que algunas aplicaciones pueden manejar mejor que otras).

La repetición de la operación de Ansible también es una opción cuando desea devolver todo el clúster a un estado completamente óptimo y, en algunos casos, puede recuperar el estado del clúster con más facilidad que con los pc. Especialmente durante una emergencia en la que el clúster está inactivo por algún motivo, una vez que todos los nodos se vuelven a ejecutar Ansible puede recuperar el clúster de forma más rápida y fiable que intentar usar los pc.

Examine el estado del clúster

Utilice pc para ver el estado del clúster.

Descripción general

Ejecutando `pcs status` Desde cualquiera de los nodos del clúster es la forma más sencilla de ver el estado general del clúster y el estado de cada recurso (como los servicios BeeGFS y sus dependencias). En esta sección se describe lo que encontrará en el resultado del `pcs status` comando.

Comprender el resultado de `pcs status`

Ejecución `pcs status` En cualquier nodo de clúster en el que se hayan iniciado los servicios de clúster (Pacemaker y Corosync). La parte superior del resultado le mostrará un resumen del clúster:

```
[root@beegfs_01 ~]# pcs status
Cluster name: hacluster
Cluster Summary:
  * Stack: corosync
  * Current DC: beegfs_01 (version 2.0.5-9.el8_4.3-ba59be7122) - partition
with quorum
  * Last updated: Fri Jul  1 13:37:18 2022
  * Last change:  Fri Jul  1 13:23:34 2022 by root via cibadmin on
beegfs_01
  * 6 nodes configured
  * 235 resource instances configured
```

En la siguiente sección se enumeran los nodos del clúster:

```
Node List:
  * Node beegfs_06: standby
  * Online: [ beegfs_01 beegfs_02 beegfs_04 beegfs_05 ]
  * OFFLINE: [ beegfs_03 ]
```

Esto indica notablemente cualquier nodo que esté en espera o sin conexión. Los nodos en espera siguen participando en el clúster, pero se marcan como no aptos para ejecutar los recursos. Los nodos que están sin conexión indican que los servicios de clúster no se están ejecutando en ese nodo, ya sea debido a que se detuvo manualmente o porque el nodo se reinició/apague.

> Cuando se inician por primera vez los nodos, se detienen los servicios del clúster y se deben iniciar manualmente para evitar que se reproduzcan accidentalmente los recursos de un nodo que no está en buen estado.

Si los nodos están en espera o sin conexión debido a un motivo no administrativo (por ejemplo, un fallo), se mostrará texto adicional junto al estado del nodo entre paréntesis. Por ejemplo, si está desactivada la delimitación y un recurso encuentra un error que verá `Node <HOSTNAME>: standby (on-fail)`. Otro estado posible es `Node <HOSTNAME>: UNCLEAN (offline)`, que se verá brevemente como un nodo está siendo vallado, pero persistirá si falló la cercado indicando que el clúster no puede confirmar el estado del nodo (esto puede bloquear que los recursos comiencen en otros nodos).

En la siguiente sección se muestra una lista de todos los recursos del clúster y sus estados:

```
Full List of Resources:
  * mgmt-monitor    (ocf::eseries:beegfs-monitor):   Started beegfs_01
  * Resource Group: mgmt-group:
    * mgmt-FS1  (ocf::eseries:beegfs-target):    Started beegfs_01
    * mgmt-IP1  (ocf::eseries:beegfs-ipaddr2):   Started beegfs_01
    * mgmt-IP2  (ocf::eseries:beegfs-ipaddr2):   Started beegfs_01
    * mgmt-service  (systemd:beegfs-mgmtd):  Started beegfs_01
[...]
```

De forma similar a los nodos, se mostrará texto adicional junto al estado del recurso entre paréntesis si hay problemas con el recurso. Por ejemplo, si Pacemaker solicita una detención de recursos y no puede completarse dentro del tiempo asignado, Pacemaker intentará cercar el nodo. Si se desactiva la delimitación o se produce un error en la operación de delimitación, el estado del recurso será `FAILED <HOSTNAME> (blocked)` Y Pacemaker no podrá iniciarlo en un nodo diferente.

Vale la pena destacar que los clusters de ha de BeeGFS utilizan una serie de agentes de recursos de OCFP personalizados optimizados de BeeGFS. En particular, el monitor BeeGFS es responsable de activar una conmutación por error cuando los recursos de BeeGFS en un nodo determinado no están disponibles.

Vuelva a configurar el clúster de alta disponibilidad y BeeGFS

Use Ansible para volver a configurar el clúster.

Descripción general

Normalmente, la reconfiguración de cualquier aspecto del clúster de alta disponibilidad de BeeGFS se debe realizar actualizando `ansible-playbook` el inventario de Ansible y volviendo a ejecutar el comando. Esto incluye la actualización de alertas, el cambio de la configuración permanente de la vallado o el ajuste de la configuración del servicio BeeGFS. Estos se ajustan utilizando el `group_vars/ha_cluster.yml` archivo y se puede encontrar una lista completa de opciones en la "Especifique la configuración de nodos de archivos comunes" sección.

Consulte a continuación para obtener más información sobre opciones de configuración seleccionadas que los administradores deben conocer al realizar tareas de mantenimiento o mantenimiento del clúster.

Cómo deshabilitar y activar delimitación

La cercado se habilita o requiere de forma predeterminada cuando se configura el clúster. En algunos casos, puede que sea conveniente desactivar temporalmente la delimitación para garantizar que los nodos no se cierren accidentalmente al realizar determinadas operaciones de mantenimiento (como la actualización del sistema operativo). Aunque esto se puede desactivar manualmente, hay sacrificios que los administradores deben tener en cuenta.

OPCIÓN 1: Desactive la esgrima con Ansible (recomendado).

Cuando se desactiva el cercado mediante Ansible, la acción en caso de fallo del monitor BeeGFS cambia de "cerca" a "en espera". Esto significa que si el monitor BeeGFS detecta un fallo, intentará poner el nodo en espera y realizar una conmutación por error de todos los servicios de BeeGFS. Fuera de la solución activa de problemas/pruebas esto es normalmente más deseable que la opción 2. La desventaja es que si un recurso no se detiene en el nodo original, se impedirá que se inicie en otro lugar (por lo que normalmente se requiere cercado para los clústeres de producción).

1. En su inventario de Ansible en `groups_vars/ha_cluster.yml` añada la siguiente configuración:

```
beegfs_ha_cluster_crm_config_options:
    stonith-enabled: False
```

2. Vuelva a ejecutar el libro de estrategia de Ansible para aplicar los cambios al clúster.

OPCIÓN 2: Desactive la delimitación manualmente.

En algunos casos puede que desee deshabilitar temporalmente la delimitación sin volver a leer Ansible, quizás para facilitar la solución de problemas o las pruebas del clúster.

> En esta configuración si el monitor BeeGFS detecta un error, el clúster intentará detener el grupo de recursos correspondiente. NO activará una conmutación por error completa ni intentará reiniciar ni mover el grupo de recursos afectado a otro host. Para recuperarse, solucione cualquier problema que se pueda ejecutar `pcs resource cleanup` o coloque manualmente el nodo en espera.

Pasos:

1. Para determinar si el cercado (stonith) está habilitado o desactivado globalmente, ejecute: `pcs property show stonith-enabled`
2. Para desactivar la secuencia de cercado: `pcs property set stonith-enabled=false`
3. Para habilitar la ejecución de cercado: `pcs property set stonith-enabled=true`

Nota: Este ajuste se anulará la próxima vez que ejecute la tableta Ansible playbook.

Actualice los componentes de clúster de alta disponibilidad

Actualice la versión de BeeGFS

Siga estos pasos para actualizar la versión BeeGFS del clúster de alta disponibilidad con

Ansible.

Descripción general

BeeGFS sigue un `major.minor.patch` esquema de control de versiones. Se proporcionan los roles de Ansible de alta disponibilidad de BeeGFS para cada `major.minor` versión compatible (p. ej., `beegfs_ha_7_2` y `beegfs_ha_7_3`). Todos los roles de alta disponibilidad se fijan a la última versión de parche de BeeGFS disponible en el momento del lanzamiento de la colección Ansible.

Ansible se debe utilizar en todas las actualizaciones de BeeGFS, incluido el desplazamiento entre versiones principales, secundarias y parches de BeeGFS. Para actualizar BeeGFS, primero tendrá que actualizar la colección BeeGFS Ansible, que también incluye las correcciones y mejoras más recientes para la automatización de la puesta en marcha/gestión y el clúster de alta disponibilidad subyacente. Incluso después de actualizar a la última versión de la colección, BeeGFS no se actualizará hasta que `ansible-playbook` se realiza con el `-e "beegfs_ha_force_upgrade=true"` configurado.

> Para obtener más información sobre las versiones de BeeGFS, consulte "Documentación de actualización de BeeGFS".

Rutas de actualización probadas

Cada versión de la colección BeeGFS se prueba con versiones específicas de BeeGFS para garantizar la interoperabilidad entre todos los componentes. También se realizan pruebas para garantizar que las actualizaciones se pueden realizar desde las versiones de BeeGFS admitidas por la última versión de la colección hasta las admitidas en la última versión.

Versión original	Actualizar la versión	MultiRail	Detalles
7.2.6	7.3.2	Sí	Actualización de la colección beegfs de v3.0.1 a v3.1.0, multirail agregó
7.2.6	7.2.8	No	Actualizando la colección beegfs de v3.0.1 a v3.1.0
7.2.8	7.3.1	Sí	Actualización mediante la colección beegfs v3.1.0, multirail añadido
7.3.1	7.3.2	Sí	Actualice utilizando beegfs Collection v3.1.0
7.3.2	7.4.1	Sí	Actualice utilizando beegfs Collection v3.2.0
7.4.1	7.4.2	Sí	Actualice utilizando beegfs Collection v3.2.0

Pasos de actualización de BeeGFS

En las siguientes secciones se ofrecen pasos para actualizar la colección Ansible BeeGFS y el propio BeeGFS. Preste especial atención a cualquier paso(s) adicional(s) para actualizar BeeGFS versiones mayores o menores.

Paso 1: Actualizar la colección BeeGFS

Para actualizaciones de colecciones con acceso a. "Galaxia de ansible", ejecute el siguiente comando:

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

Para actualizaciones de colecciones sin conexión, descargue la colección desde "Galaxia de ansible" haciendo clic en el deseado `Install Version` y después `Download tarball`. Transfiera el tarball al nodo de control de Ansible y ejecute el siguiente comando.

```
ansible-galaxy collection install netapp_eseries-beegfs-<VERSION>.tar.gz
--upgrade
```

Consulte "Instalando colecciones" si quiere más información.

Paso 2: Actualice el inventario de Ansible

Realice las actualizaciones necesarias o deseadas que necesite en los archivos de inventario de Ansible del clúster. Consulte la "Notas sobre la actualización de la versión" siguiente sección para obtener información detallada sobre sus requisitos de actualización específicos. Consulta la "Descripción general del inventario de Ansible" sección para obtener información general sobre la configuración de tu inventario de BeeGFS HA.

Paso 3: Actualizar Ansible playbook (cuando se actualizan solo versiones principales o secundarias)

Si va a cambiar entre versiones principal o secundaria, en `playbook.yml` el archivo usado para implementar y mantener el clúster, actualice el nombre `beegfs_ha_<VERSION>` del rol para reflejar la versión deseada. Por ejemplo, si desea desplegar BeeGFS 7,4, esto sería `beegfs_ha_7_4`:

```
- hosts: all
  gather_facts: false
  any_errors_fatal: true
  collections:
    - netapp_eseries.beegfs
  tasks:
    - name: Ensure BeeGFS HA cluster is setup.
      ansible.builtin.import_role:  # import_role is required for tag
availability.
        name: beegfs_ha_7_4
```

Si desea obtener más información sobre el contenido de este archivo de playbook, consulte "Ponga en marcha el clúster de alta disponibilidad de BeeGFS"la sección.

Paso 4: Ejecute la actualización de BeeGFS

Para aplicar la actualización de BeeGFS:

```
ansible-playbook -i inventory.yml beegfs_ha_playbook.yml -e
"beegfs_ha_force_upgrade=true" --tags beegfs_ha
```

Entre bastidores, el rol de BeeGFS ha se encargará de:

- Asegúrese de que el clúster esté en estado óptimo en cada servicio BeeGFS ubicado en su nodo preferido.

- Ponga el clúster en modo de mantenimiento.
- Actualice los componentes del clúster de alta disponibilidad (si es necesario).
- Actualice cada nodo de archivo de uno en uno de los siguientes modos:
  - Colóquela en espera y realice la conmutación al nodo de respaldo de sus servicios en el nodo secundario.
  - Actualizar paquetes BeeGFS.
  - Servicios de respaldo.
- Mueva el clúster fuera del modo de mantenimiento.

Notas de actualización de la versión

Actualización desde BeeGFS versión 7.2.6 o 7.3.0

Cambios en la autenticación basada en conexión

Las versiones de BeeGFS publicadas después de 7.3.1 dejarán de permitir que los servicios se inicien sin especificar ni un `connAuthFile` o ajuste `connDisableAuthentication=true` en el archivo de configuración del servicio. Se recomienda encarecidamente habilitar la seguridad de autenticación basada en conexión. Consulte "Autenticación basada en conexión BeeGFS" si quiere más información.

De forma predeterminada, la `beegfs_ha*` Los roles generarán y distribuirán este archivo, añadiendo también al nodo de control Ansible en `<playbook_directory>/files/beegfs/<beegfs_mgmt_ip_address>_connAuthFile`. La `beegfs_client` el rol también comprobará la presencia de este archivo y lo suministrará a los clientes si está disponible.

> Si la `beegfs_client` la función no se ha utilizado para configurar clientes, este archivo deberá distribuirse manualmente a cada cliente y a la `connAuthFile` de la `beegfs-client.conf` conjunto de archivos para utilizarlo. Al actualizar desde una versión anterior de BeeGFS en la que la autenticación basada en conexión no estaba activada, los clientes perderán el acceso a menos que la autenticación basada en conexión esté deshabilitada como parte de la configuración de actualización `beegfs_ha_conn_auth_enabled: false` pulg `group_vars/ha_cluster.yml` (no recomendado).

Para obtener más detalles y opciones de configuración alternativas, consulte el paso para configurar la autenticación de conexión en la "Especifique la configuración de nodos de archivos comunes" sección.

Actualice la cabina de almacenamiento E-Series

Siga estos pasos para actualizar las cabinas de almacenamiento E-Series del clúster de alta disponibilidad (nodos de bloque).

Descripción general

Mantener las cabinas de almacenamiento de NetApp E-Series del clúster de alta disponibilidad actualizadas con el firmware más reciente garantiza un rendimiento óptimo y una mayor seguridad. Las actualizaciones de firmware para la cabina de almacenamiento se aplican a través de archivos de sistema operativo SANtricity, NVSRAM y firmware de la unidad.

> Aunque las cabinas de almacenamiento se pueden actualizar con el clúster de alta disponibilidad en línea, se recomienda colocar el clúster en modo de mantenimiento para todas las actualizaciones.

Pasos de actualización del nodo de bloque

Los siguientes pasos describen cómo actualizar el firmware de las cabinas de almacenamiento mediante `Netapp_Eseries.Santricity` la colección Ansible. Antes de continuar, revise "Consideraciones de renovación"para actualizar los sistemas E-Series.

> Solo es posible actualizar a SANtricity OS 11,80 o versiones posteriores desde 11.70.5P1. Primero, la cabina de almacenamiento debe actualizarse a 11.70.5P1 antes de aplicar nuevas actualizaciones.

1. Valide que el nodo de control de Ansible utilice la colección de Ansible más reciente de SANtricity.
   - Para actualizaciones de colecciones con acceso a. "Galaxia de ansible", ejecute el siguiente comando:

     ```
     ansible-galaxy collection install netapp_eseries.santricity --upgrade
     ```

   - Para actualizaciones sin conexión, descargue el tarball de recopilación de "Galaxia de ansible", transfiéralo al nodo de control y ejecute:

     ```
     ansible-galaxy collection install netapp_eseries-santricity-
     <VERSION>.tar.gz --upgrade
     ```

     Consulte "Instalando colecciones" si quiere más información.

2. Obtenga el firmware más reciente para la cabina de almacenamiento y las unidades.
   a. Descargue los archivos de firmware.
      - **Sistema operativo SANtricity y NVSRAM:** Navegue hasta la "Sitio de soporte de NetApp"versión más reciente de SANtricity OS y NVSRAM para su modelo de cabina de almacenamiento y descargue la versión más reciente de NVSRAM.
      - **Firmware de la unidad:** Navegue hasta "Sitio del firmware de un disco E-Series"y descargue el firmware más reciente para cada uno de los modelos de unidad de la cabina de almacenamiento.

   b. Almacene los archivos de firmware del sistema operativo SANtricity, NVSRAM y la unidad en `<inventory_directory>/packages` el directorio del nodo de control Ansible.
3. Si es necesario, actualice los archivos de inventario de Ansible del clúster para incluir todas las cabinas de almacenamiento (nodos de bloque) que requieran actualizaciones. Para obtener orientación, consulte la "Descripción general del inventario de Ansible" sección.
4. Compruebe que el clúster tenga un estado óptimo y cada servicio BeeGFS se encuentre en su nodo preferido. Consulte "Examine el estado del clúster" para obtener más información.
5. Coloque el clúster en modo de mantenimiento siguiendo las instrucciones que se indican en "Coloque el clúster en modo de mantenimiento".
6. Cree un nuevo libro de estrategia de Ansible llamado `update_block_node_playbook.yml`. Complete el libro de estrategia con el siguiente contenido, sustituyendo las versiones del sistema operativo

SANtricity, NVSRAM y firmware de la unidad a la ruta de actualización que desee:

```
- hosts: eseries_storage_systems
  gather_facts: false
  any_errors_fatal: true
  collections:
    - netapp_eseries.santricity
  vars:
    eseries_firmware_firmware: "packages/<SantricityOS>.dlp"
    eseries_firmware_nvsram: "packages/<NVSRAM>.dlp"
    eseries_drive_firmware_firmware_list:
      - "packages/<drive_firmware>.dlp"
    eseries_drive_firmware_upgrade_drives_online: true

  tasks:
    - name: Configure NetApp E-Series block nodes.
      import_role:
        name: nar_santricity_management
```

7. Para iniciar las actualizaciones, ejecute el siguiente comando desde el nodo de control de Ansible:

```
ansible-playbook -i inventory.yml update_block_node_playbook.yml
```

8. Una vez que se complete el libro de estrategia, compruebe que cada cabina de almacenamiento tenga el estado Óptimo.
9. Mueva el clúster fuera del modo de mantenimiento y compruebe que el clúster tenga el estado óptimo con cada servicio BeeGFS en su nodo preferido.

Mantenimiento y mantenimiento

Servicios de conmutación por error y conmutación tras recuperación

Desplazamiento de servicios BeeGFS entre nodos del clúster.

Descripción general

Los servicios BeeGFS pueden realizar una conmutación por error entre los nodos del clúster para garantizar que los clientes puedan continuar accediendo al sistema de archivos si un nodo experimenta un error o necesita realizar tareas de mantenimiento planificadas. En esta sección se describen distintas formas en las que los administradores pueden recuperar el clúster después de un fallo o mover manualmente servicios entre nodos.

Pasos

Conmutación al respaldo (planificada)

En general, cuando necesite desconectar un solo nodo de archivos para realizar el mantenimiento, querrá mover (o drenar) todos los servicios de BeeGFS de ese nodo. Esto se puede lograr poniendo el nodo en espera en primer lugar:

```
pcs node standby <HOSTNAME>
```

Después de verificar utilizando `pcs status` todos los recursos se han reiniciado en el nodo de archivos alternativo, puede apagar o realizar otros cambios en el nodo según sea necesario.

Conmutación tras recuperación (después de una conmutación al respaldo planificada)

Cuando esté listo para restaurar los servicios BeeGFS en el nodo preferido, ejecute primero `pcs status` Y verifique en la "Lista de nodos" el estado es en espera. Si el nodo se reinició, aparecerá sin conexión hasta que los servicios del clúster estén en línea:

```
pcs cluster start <HOSTNAME>
```

Una vez que el nodo esté en línea, salga del modo de espera con:

```
pcs cluster node unstandby <HOSTNAME>
```

Por último, reubique todos los servicios de BeeGFS en sus nodos preferidos con:

```
pcs resource relocate run
```

Conmutación tras recuperación (después de una conmutación al respaldo no planificada)

Si un nodo experimenta un fallo de hardware o de otro tipo, el clúster de alta disponibilidad debería reaccionar automáticamente y mover sus servicios a un nodo en buen estado, lo que proporciona tiempo para que los administradores tomen acciones correctivas. Antes de continuar, consulte la "resolución de problemas" sección para determinar la causa de la conmutación por error y resolver los problemas pendientes. Una vez que el nodo se vuelve a encender y en buen estado, puede continuar con la conmutación tras recuperación.

Cuando un nodo se arranca tras un reinicio no planificado (o planificado), los servicios de clúster no se establecen para iniciarse automáticamente, por lo que primero tendrá que conectar el nodo con:

```
pcs cluster start <HOSTNAME>
```

A continuación, borre los errores de los recursos y restablezca el historial de cercas del nodo:

```
pcs resource cleanup node=<HOSTNAME>
pcs stonith history cleanup <HOSTNAME>
```

Verifique en `pcs status` el nodo está en línea y en buen estado. De forma predeterminada, los servicios de BeeGFS no se podrán recuperar automáticamente para evitar que los recursos vuelvan a un nodo que no esté en buenas estado. Cuando esté listo, devuelva todos los recursos del clúster a los nodos preferidos con:

```
pcs resource relocate run
```

Mover servicios de BeeGFS individuales a nodos de archivo alternativos

Mueva permanentemente un servicio BeeGFS a un nuevo nodo de archivo

Si desea cambiar de forma permanente el nodo de archivo preferido de un servicio BeeGFS individual, ajuste el inventario de Ansible para ver primero el nodo preferido y volver a ejecutar el libro de estrategia de Ansible.

Por ejemplo, en este archivo de ejemplo `inventory.yml` , beegfs_01 es el nodo de archivos preferido para ejecutar el servicio de gestión BeeGFS:

```
        mgmt:
          hosts:
            beegfs_01:
            beegfs_02:
```

Si se invierte el pedido, se preferirían los servicios de gestión en beegfs_02:

```
        mgmt:
          hosts:
            beegfs_02:
            beegfs_01:
```

Mueva temporalmente un servicio BeeGFS a otro nodo de archivo

Generalmente, si un nodo está en proceso de mantenimiento, deberá utilizar los [pasos de conmutación por error y conmutación por recuperación](#failover-and-failback) para mover todos los servicios fuera de ese nodo.

Si por algún motivo necesita mover un servicio individual a un nodo de archivo diferente ejecutado:

```
pcs resource move <SERVICE>-monitor <HOSTNAME>
```

> No especifique recursos individuales ni el grupo de recursos. Especifique siempre el nombre del monitor para el servicio BeeGFS que desea reubicar. Por ejemplo, para mover el servicio de gestión BeeGFS a beegfs_02 ejecute: `pcs resource move mgmt-monitor beegfs_02`. Este proceso se puede repetir para mover uno o varios servicios de sus nodos preferidos. Verifique que `pcs status` los servicios se han reubicado/iniciado en el nuevo nodo.

Para devolver un servicio BeeGFS a su nodo preferido, borre primero las restricciones de recursos temporales (repita este paso según sea necesario para varios servicios):

```
pcs resource clear <SERVICE>-monitor
```

A continuación, cuando esté listo para mover realmente los servicios de nuevo a sus nodos preferidos ejecutar:

```
pcs resource relocate run
```

Nota este comando reubicará los servicios que ya no tengan restricciones temporales de recursos que no estén en sus nodos preferidos.

Coloque el clúster en modo de mantenimiento

Evite que el clúster de alta disponibilidad reaccione accidentalmente a los cambios previstos del entorno.

Descripción general

Si pone el clúster en el modo de mantenimiento, se deshabilita toda la supervisión de recursos y se impide que Pacemaker mueva o gestione los recursos del clúster de algún otro modo. Todos los recursos permanecerán en ejecución en sus nodos originales, independientemente de que haya una condición de fallo temporal que impida que se pueda acceder a ellos. Los escenarios en los que esto es recomendable/útil incluyen:

- Mantenimiento de red que puede interrumpir temporalmente las conexiones entre nodos de archivo y servicios BeeGFS.
- Actualizaciones de nodos de bloques.
- Actualizaciones del sistema operativo, el kernel u otros paquetes del nodo de archivos.

Por lo general, el único motivo para poner manualmente el clúster en modo de mantenimiento es impedir que este reaccione a cambios externos en el entorno. Si un nodo individual del clúster requiere reparación física no utilice modo de mantenimiento y simplemente coloque ese nodo en espera tras el procedimiento anterior. Tenga en cuenta que el nuevo enrutamiento de Ansible pondrá automáticamente el clúster en modo de mantenimiento para facilitar la mayoría de las tareas de mantenimiento de software, incluidas las actualizaciones y los cambios de configuración.

Pasos

Para comprobar si el clúster se encuentra en modo de mantenimiento ejecutar:

```
pcs property show maintenance-mode
```

Esto devolverá el valor false cuando el clúster esté funcionando normalmente. Para habilitar la ejecución del modo de mantenimiento:

```
pcs property set maintenance-mode=true
```

Puede verificar ejecutando el estado del pc y asegurando que todos los recursos muestren "(no administrado)". Para desconectar el clúster del modo de mantenimiento ejecute:

```
pcs property set maintenance-mode=false
```

Detenga e inicie el clúster

Detener e iniciar correctamente el clúster de alta disponibilidad.

Descripción general

En esta sección se describe cómo apagar y reiniciar correctamente el clúster BeeGFS. Algunos ejemplos de casos en los que esto puede ser necesario son el mantenimiento eléctrico o la migración entre centros de datos o racks.

Pasos

Si por algún motivo necesita detener todo el clúster BeeGFS y apagar todos los servicios que se ejecutan:

```
pcs cluster stop --all
```

También es posible detener el clúster en nodos individuales (que automáticamente conmutarán por error los servicios a otro nodo), aunque se recomienda poner primero el nodo en espera (consulte la "conmutación al respaldo" sección):

```
pcs cluster stop <HOSTNAME>
```

Para iniciar los servicios y recursos del clúster en todos los nodos ejecutados:

```
pcs cluster start --all
```

O inicie servicios en un nodo específico con:

```
pcs cluster start <HOSTNAME>
```

En este momento, corre `pcs status` Compruebe que el clúster y los servicios BeeGFS se inicien en todos los nodos y que los servicios se estén ejecutando en los nodos que espera.

> Según el tamaño del clúster, puede tardar en detenerse de algún momento (segundos a minutos) para que todo el clúster se detenga o se muestre iniciado en `pcs status`. Si `pcs cluster <COMMAND>` Se bloquea durante más de cinco minutos antes de ejecutar "Ctrl+C" para cancelar el comando, iniciar sesión en cada nodo del clúster y utilizar `pcs status` Para ver si los servicios de clúster (Corosync/Pacemaker) aún se están ejecutando en ese nodo. Desde cualquier nodo en el que el clúster siga estando activo, puede comprobar qué recursos están bloqueando el clúster. Solucione manualmente el problema y el comando debería estar completo o se puede volver a ejecutar para detener el resto de servicios.

Sustituya los nodos de archivo

Reemplazar un nodo de archivo si el servidor original está defectuoso.

Descripción general

Esta es una descripción general de los pasos necesarios para reemplazar un nodo de archivo del clúster. Estos pasos suponen un error en el nodo de archivo debido a un problema de hardware y se reemplazaron por un nuevo nodo de archivo idéntico.

Pasos:

1. Sustituya físicamente el nodo de archivo y restaure todo el cableado al nodo de bloque y a la red de almacenamiento.
2. Vuelva a instalar el sistema operativo en el nodo de archivos, incluida la adición de suscripciones a Red Hat.
3. Configure la gestión y las redes de BMC en el nodo de archivo.
4. Actualice el inventario de Ansible si el nombre de host, la IP, las asignaciones de interfaz de PCIe a lógica o cualquier otro cambio relacionado con el nodo de archivo nuevo. Generalmente, esto no es necesario si el nodo se reemplazó con un hardware de servidor idéntico y si utiliza la configuración de red original.
   a. Por ejemplo, si el nombre de host ha cambiado, cree (o cambie de nombre) el archivo de inventario del nodo (`host_vars/<NEW_NODE>.yml`) Luego en el archivo de inventario de Ansible (`inventory.yml`), reemplace el nombre del nodo antiguo por el nuevo nombre del nodo:

```
all:
    ...
    children:
    ha_cluster:
        children:
        mgmt:
            hosts:
            node_h1_new:   # Replaced "node_h1" with "node_h1_new"
            node_h2:
```

5. De uno de los otros nodos del clúster, quite el nodo antiguo: `pcs cluster node remove <HOSTNAME>`.

NO CONTINÚE ANTES DE EJECUTAR ESTE PASO.

6. En el nodo de control Ansible:
   a. Quite la clave SSH antigua con:

```
`ssh-keygen -R <HOSTNAME_OR_IP>`
```

   b. Configure SSH sin contraseña al nodo de sustitución con:

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

7. Vuelva a ejecutar el libro de estrategia de Ansible para configurar el nodo y añadirlo al clúster:

```
ansible-playbook -i <inventory>.yml <playbook>.yml
```

8. En este momento, corre `pcs status` y compruebe que el nodo sustituido ahora aparece y ejecuta servicios.

Expanda o reduzca el clúster

Añada o quite bloques de creación del clúster.

Descripción general

En esta sección se documentan varias consideraciones y opciones para ajustar el tamaño del clúster de alta disponibilidad de BeeGFS. Normalmente, el tamaño del clúster se ajusta agregando o quitando elementos básicos, que normalmente son dos nodos de archivo configurados como un par de alta disponibilidad. También es posible agregar o quitar nodos de archivos individuales (u otros tipos de nodos de clústeres) si es necesario.

Añadir una elemento básico al clúster

Consideraciones

El crecimiento del clúster mediante la adición de elementos básicos adicionales es un proceso sencillo. Antes de empezar a tener en cuenta las restricciones en torno al número mínimo y máximo de nodos de clúster en cada clúster de alta disponibilidad, y determinar si debe añadir nodos al clúster de alta disponibilidad existente o crear un nuevo clúster de alta disponibilidad. Normalmente, cada bloque de creación consta de dos nodos de archivo, pero tres nodos son el número mínimo de nodos por clúster (para establecer quórum), y diez son el máximo recomendado (probado). En situaciones avanzadas es posible añadir un solo nodo "tiebreaker" que no ejecute ningún servicio BeeGFS al poner en marcha un clúster de dos nodos. Póngase en contacto con el servicio de soporte de NetApp si está considerando realizar dicha implementación.

Tenga en cuenta estas restricciones y cualquier crecimiento futuro del clúster anticipado al decidir cómo ampliar el clúster. Por ejemplo, si tiene un clúster de seis nodos y necesita añadir cuatro nodos más, se recomienda simplemente iniciar un nuevo clúster de alta disponibilidad.

Recuerde que un solo sistema de archivos BeeGFS puede consistir en varios clústeres de alta disponibilidad independientes. Esto permite que los sistemas de archivos sigan escalando entre los límites de hardware recomendado y los componentes de clúster de alta disponibilidad subyacentes.

Pasos

Al agregar un elemento básico al clúster, deberá crear los `host_vars` archivos para cada uno de los nuevos nodos de archivo y los nodos de bloque (cabinas E-Series). Los nombres de estos hosts deben agregarse al inventario, junto con los nuevos recursos que se van a crear. `group_vars`Se deberán crear los archivos correspondientes para cada nuevo recurso. Consulte la "utilizar arquitecturas personalizadas" sección para obtener más información.

Una vez creados los archivos correctos, todo lo que se necesita es volver a ejecutar la automatización mediante el comando:

```
ansible-playbook -i <inventory>.yml <playbook>.yml
```

Eliminación de un elemento básico del clúster

Hay una serie de consideraciones que tener en cuenta cuando se necesita retirar un bloque de construcción, por ejemplo:

- ¿Qué servicios BeeGFS se están ejecutando en este bloque de creación?
- ¿Solo se retiran los nodos de archivos y los nodos de bloque deben adjuntarse a nuevos nodos de archivos?
- Si se retira todo el elemento básico, ¿deben moverse los datos a un nuevo elemento básico, dispersarse en nodos existentes en el clúster o moverse a un nuevo sistema de archivos BeeGFS u otro sistema de almacenamiento?
- ¿Puede suceder esto durante una interrupción o se debe realizar de forma no disruptiva?
- ¿El elemento básico se está utilizando de forma activa o contiene principalmente datos que ya no están activos?

Debido a la diversidad de puntos de partida posibles y a los estados finales deseados, póngase en contacto con el soporte de NetApp para que podamos identificar y ayudar a implementar la mejor estrategia en función de su entorno y sus requisitos.

Solucionar problemas

Solucionar problemas de un clúster de alta disponibilidad de BeeGFS.

Descripción general

En esta sección se explica cómo investigar y solucionar varios fallos y otros escenarios que pueden surgir al utilizar un clúster de alta disponibilidad de BeeGFS.

Guías de solución de problemas

Investigando conmutaciones al respaldo inesperadas

Cuando un nodo tiene una barrera aplicada de forma inesperada y sus servicios pasan a otro nodo, el primer paso debe ser comprobar si el clúster indica algún error de recurso en la parte inferior de `pcs status`. Por lo general, no habrá nada presente si la delimitación se ha realizado correctamente y se han reiniciado los recursos en otro nodo.

Por lo general, el siguiente paso será buscar a través de los registros del sistema utilizando `journalctl` En cualquiera de los nodos de archivo restantes (los registros de Pacemaker se sincronizan en todos los nodos). Si sabe la hora en que ocurrió el fallo, puede iniciar la búsqueda justo antes de que se produjera el fallo (generalmente se recomienda al menos diez minutos antes):

```
journalctl --since "<YYYY-MM-DD HH:MM:SS>"
```

En las siguientes secciones se muestra el texto común que se puede obtener en los registros para delimitar aún más la investigación.

Pasos para investigar/resolver

Paso 1: Compruebe si el monitor BeeGFS ha detectado un fallo:

Si el monitor BeeGFS ha activado la conmutación por error, debería aparecer un error (si no es así, continúe con el siguiente paso).

```
journalctl --since "<YYYY-MM-DD HH:MM:SS>" | grep -i unexpected
[...]
Jul 01 15:51:03 beegfs_01 pacemaker-schedulerd[9246]:  warning: Unexpected
result (error: BeeGFS service is not active!) was recorded for monitor of
meta_08-monitor on beegfs_02 at Jul  1 15:51:03 2022
```

En esta instancia, el servicio de BeeGFS meta_08 se detuvo por algún motivo. Para continuar con la solución de problemas, debemos iniciar beegfs_02 y revisar los registros del servicio en `/var/log/beegfs-meta-meta_08_tgt_0801.log`. Por ejemplo, el servicio BeeGFS puede haber encontrado un error de aplicación debido a un problema interno o al nodo.

> A diferencia de los registros de Pacemaker, los registros de los servicios BeeGFS no se distribuyen a todos los nodos del clúster. Para investigar estos tipos de errores, son necesarios los registros del nodo original en el que se produjo el error.

Entre los posibles problemas que podría generar el monitor se incluyen los siguientes:

- No se puede acceder a los destinos.
  - Descripción: Indica que no se puede acceder a los volúmenes de bloques.
  - Solución de problemas:
    - Si también no se pudo iniciar el servicio en el nodo de archivos alternativo, confirme que el nodo de bloque está en buen estado.

        - Compruebe si hay problemas físicos que impidan el acceso a los nodos de bloque desde este nodo de archivos, por ejemplo, adaptadores o cables InfiniBand defectuosos.
- ¡No se puede acceder a la red!
    - Descripción: Ninguno de los adaptadores utilizados por los clientes para conectarse a este servicio BeeGFS estaba en línea.
    - Solución de problemas:
        - Si varios o todos los nodos de archivo se vieron afectados, compruebe si se produjo un error en la red utilizada para conectar los clientes BeeGFS y el sistema de archivos.
        - Compruebe si hay problemas físicos que impidan el acceso a los clientes desde este nodo de archivos, por ejemplo, adaptadores o cables InfiniBand defectuosos.
- El servicio BeeGFS no está activo.
    - Descripción: Un servicio BeeGFS se ha detenido inesperadamente.
    - Solución de problemas:
        - En el nodo de archivo que notificó el error, compruebe los registros del servicio BeeGFS afectado para ver si se produjo un fallo. Si esto sucede, abra un caso con el soporte de NetApp para que pueda investigar el bloqueo.
        - Si no se ha informado de errores en el registro de BeeGFS, compruebe los registros del diario para ver si systemd ha registrado un motivo por el que se ha detenido el servicio. En algunos casos, es posible que el servicio BeeGFS no haya recibido la oportunidad de registrar ningún mensaje antes de que se terminara el proceso (por ejemplo, si se ejecutó a alguien) `kill -9 <PID>`).

Paso 2: Compruebe si el nodo dejó el clúster de forma inesperada

En caso de que el nodo sufriera un error de hardware catastrófico (por ejemplo, murió la placa del sistema) o se produjo un error de alerta en el kernel o un problema de software similar, el monitor BeeGFS no informará de este error. En su lugar, busque el nombre de host y debería ver mensajes del Pacemaker que indican que el nodo se ha perdido inesperadamente:

```
journalctl --since "<YYYY-MM-DD HH:MM:SS>" | grep -i <HOSTNAME>
[...]
Jul 01 16:18:01 beegfs_01 pacemaker-attrd[9245]:  notice: Node beegfs_02
state is now lost
Jul 01 16:18:01 beegfs_01 pacemaker-controld[9247]:  warning:
Stonith/shutdown of node beegfs_02 was not expected
```

Paso 3: Verifique que Pacemaker pudo cercar el nodo

En todos los escenarios debería ver que Pacemaker intenta cercar el nodo para verificar que está realmente sin conexión (los mensajes exactos pueden variar según la causa del cercado):

```
Jul 01 16:18:02 beegfs_01 pacemaker-schedulerd[9246]:  warning: Cluster
node beegfs_02 will be fenced: peer is no longer part of the cluster
Jul 01 16:18:02 beegfs_01 pacemaker-schedulerd[9246]:  warning: Node
beegfs_02 is unclean
Jul 01 16:18:02 beegfs_01 pacemaker-schedulerd[9246]:  warning: Scheduling
Node beegfs_02 for STONITH
```

Si la acción de cercado se completa correctamente, verá mensajes como:

```
Jul 01 16:18:14 beegfs_01 pacemaker-fenced[9243]:  notice: Operation 'off'
[2214070] (call 27 from pacemaker-controld.9247) for host 'beegfs_02' with
device 'fence_redfish_2' returned: 0 (OK)
Jul 01 16:18:14 beegfs_01 pacemaker-fenced[9243]:  notice: Operation 'off'
targeting beegfs_02 on beegfs_01 for pacemaker-
controld.9247@beegfs_01.786df3a1: OK
Jul 01 16:18:14 beegfs_01 pacemaker-controld[9247]:  notice: Peer
beegfs_02 was terminated (off) by beegfs_01 on behalf of pacemaker-
controld.9247: OK
```

Si la acción de delimitación falló por algún motivo, los servicios BeeGFS no podrán reiniciarse en otro nodo para evitar el riesgo de corrupción de datos. Eso sería un problema para investigar por separado, si, por ejemplo, el dispositivo de cercado (PDU o BMC) era inaccesible o mal configurado.

Acciones de recursos fallidos de direcciones (que se encuentran en la parte inferior del estado de los pc)

Si falla un recurso necesario para ejecutar un servicio BeeGFS, el monitor BeeGFS activará una conmutación por error. Si esto ocurre, es probable que no haya ninguna lista de acciones de recursos fallidas en la parte inferior de `pcs status` y debe consultar los pasos sobre cómo "conmutación tras recuperación tras fallos no planificada".

De lo contrario, normalmente sólo deberían haber dos escenarios en los que verá "acciones de recursos fallidas".

Pasos para investigar/resolver

Escenario 1: Se detectó un problema temporal o permanente con un agente de esgrima y se reinició u movió a otro nodo.

Algunos agentes de cercado son más confiables que otros, y cada uno implementará su propio método de monitoreo para garantizar que el dispositivo de cercado esté listo. En particular, el agente de esgrima de Redfish ha sido visto para informar de acciones de recursos fallidas como las siguientes, aunque todavía se muestre iniciado:

```
  * fence_redfish_2_monitor_60000 on beegfs_01 'not running' (7):
call=2248, status='complete', exitreason='', last-rc-change='2022-07-26
08:12:59 -05:00', queued=0ms, exec=0ms
```

No se espera que un agente de delimitación que informe sobre acciones de recursos fallidas en un determinado nodo active una conmutación por error de los servicios BeeGFS que se ejecutan en ese nodo. Solo hay que reiniciar automáticamente en un mismo nodo o en uno distinto.

Pasos para resolver:

1. Si el agente de cercado se niega sistemáticamente a ejecutarse en todos los nodos o en un subconjunto de ellos, compruebe si dichos nodos pueden conectarse al agente de cercado y compruebe que el agente de cercado esté configurado correctamente en el inventario de Ansible.
   a. Por ejemplo, si un agente de cercado Redfish (BMC) se está ejecutando en el mismo nodo que es responsable de cercado, y la gestión del SO y las IP de BMC están en la misma interfaz física, algunas configuraciones de switches de red no permitirán la comunicación entre las dos interfaces (para evitar bucles de red). De forma predeterminada, el clúster de alta disponibilidad intentará evitar colocar agentes de cercado en el nodo que sean responsables de cercado, pero esto puede suceder en algunos escenarios/configuraciones.
2. Una vez que se resuelven todos los problemas (o si el problema parece efímero), ejecute `pcs resource cleanup` para restablecer las acciones de recursos fallidas.

Escenario 2: El monitor BeeGFS detectó un problema y activó un fallo, pero por algún motivo los recursos no se pudieron iniciar en un nodo secundario.

Siempre que la delimitación esté habilitada y que el recurso no se haya bloqueado para detenerse en el nodo original (consulte la sección de solución de problemas "standby (on-fail)"), los motivos más probables incluyen problemas para iniciar el recurso en un nodo secundario debido a lo siguiente:

- El nodo secundario ya estaba desconectado.
- Un problema de configuración física o lógica impidió que el secundario acceda a los volúmenes de bloques utilizados como destinos de BeeGFS.

Pasos para resolver:

1. Para cada entrada de las acciones de recursos fallidas:
   a. Confirme que la acción de recurso fallida fue una operación de inicio.
   b. Según el recurso indicado y el nodo especificado en las acciones de recursos con errores:
      i. Busque y corrija los problemas externos que podrían impedir que el nodo inicie el recurso especificado. Por ejemplo, si no se pudo iniciar la dirección IP de BeeGFS (IP flotante), compruebe que al menos una de las interfaces necesarias está conectada/conectada y cableada al conmutador de red correcto. Si se produce un error en un objetivo de BeeGFS (dispositivo de bloque/volumen de E-Series), compruebe que las conexiones físicas con los nodos de bloque back-end estén conectadas según lo esperado y verifique que los nodos de bloque estén en buen estado.
   c. Si no hay problemas externos obvios y desea un motivo raíz para este incidente, se recomienda abrir un caso con la compatibilidad de NetApp para investigar antes de continuar, ya que los siguientes pasos pueden hacer que sea un desafío/imposible el análisis de causa raíz (RCA).
2. Después de resolver cualquier problema externo:
   a. Comente cualquier nodo no funcional del archivo Ansible Inventory.yml y vuelva a ejecutar el libro de estrategia de Ansible completo para garantizar que toda la configuración lógica se configure correctamente en los nodos secundarios.
      i. Nota: No olvide dejar de comentar estos nodos y volver a ejecutar la tableta playbook una vez que el estado de los nodos sea bueno y esté listo para realizar la conmutación tras recuperación.

b. También puede intentar recuperar manualmente el clúster:

   i. Vuelva a colocar todos los nodos sin conexión en línea mediante: `pcs cluster start <HOSTNAME>`

   ii. Borre todas las acciones de recursos fallidas mediante: `pcs resource cleanup`

   iii. Ejecute el estado del pc y verifique que todos los servicios comiencen según lo esperado.

   iv. Si es necesario, corre `pcs resource relocate run` para devolver los recursos a su nodo preferido (si está disponible).

Cuestiones comunes

Los servicios de BeeGFS no realizan una conmutación por error ni una conmutación tras recuperación cuando se le solicite

Asunto probable: la `pcs resource relocate` se ejecutó el comando de ejecución, pero nunca se terminó correctamente.

Cómo comprobar: Ejecutar `pcs constraint --full` Y compruebe si existen restricciones de ubicación con un ID de `pcs-relocate-<RESOURCE>`.

Cómo resolver: Ejecutar `pcs resource relocate clear` a continuación, vuelva a ejecutar `pcs constraint --full` para verificar que se han eliminado las restricciones adicionales.

Un nodo en el estado del pc muestra "standby (on-fail)" cuando está desactivado el cercado

Problema probable: Pacemaker no pudo confirmar con éxito todos los recursos fueron detenidos en el nodo que falló.

Cómo resolver:

1. Ejecución `pcs status` y busque los recursos que no se "hayan iniciado" o que muestren errores en la parte inferior del resultado y resuelva cualquier problema.
2. Para volver a poner en línea el nodo `pcs resource cleanup --node=<HOSTNAME>`.

Después de una conmutación por error inesperada, los recursos muestran "iniciado (en caso de fallo)" en el estado de los pc cuando se activa la delimitación

Problema probable: se produjo Un problema que provocó una conmutación por error, pero Pacemaker no pudo verificar que el nodo estaba vallado. Esto podría ocurrir porque la delimitación estaba mal configurada o hubo un problema con el agente de cercado (ejemplo: La PDU se desconectó de la red).

Cómo resolver:

1. Compruebe que el nodo esté apagado.

   > Si el nodo que especifique no está apagado pero si ejecuta servicios o recursos del clúster, se producirán errores en los datos o en el clúster.

2. Confirmar manualmente la esgrima con: `pcs stonith confirm <NODE>`

En este punto, los servicios deben terminar de conmutar por error y reiniciarse en otro nodo en buen estado.

Tareas comunes de solución de problemas

Reinicie los servicios BeeGFS individuales

Normalmente, si es necesario reiniciar un servicio BeeGFS (por ejemplo, para facilitar un cambio en la configuración), debe hacerlo actualizando el inventario de Ansible y volviendo a ejecutar el libro de estrategia. En algunos casos, puede que sea conveniente reiniciar servicios individuales para facilitar la solución de problemas más rápida, por ejemplo, cambiar el nivel de registro sin tener que esperar a que se ejecute el libro de estrategia completo.

> A menos que también se añadan cambios manuales al inventario de Ansible, se revertirá la próxima vez que se ejecute el libro de estrategia de Ansible.

Opción 1: Reinicio controlado por sistema

Si existe un riesgo de que el servicio BeeGFS no se reinicie correctamente con la nueva configuración, coloque primero el clúster en modo de mantenimiento para evitar que el monitor BeeGFS detecte que el servicio se detiene y active una conmutación por error no deseada:

```
pcs property set maintenance-mode=true
```

Si es necesario, realice cualquier cambio en la configuración de servicios en `/mnt/<SERVICE_ID>/_config/beegfs-.conf` (ejemplo: `/mnt/meta_01_tgt_0101/metadata_config/beegfs-meta.conf`) a continuación, utilice systemd para reiniciarlo:

```
systemctl restart beegfs-*@<SERVICE_ID>.service
```

Ejemplo: `systemctl restart beegfs-meta@meta_01_tgt_0101.service`

Opción 2: Reinicio controlado por marcapasos

Si no le preocupa la nueva configuración, puede hacer que el servicio se detenga de forma inesperada (por ejemplo, simplemente cambiando el nivel de registro), o está en una ventana de mantenimiento y no le preocupa el tiempo de inactividad, puede reiniciar el monitor BeeGFS para el servicio que desea reiniciar:

```
pcs resource restart <SERVICE>-monitor
```

Por ejemplo, para reiniciar el servicio de gestión de BeeGFS: `pcs resource restart mgmt-monitor`

Avisos legales

Los avisos legales proporcionan acceso a las declaraciones de copyright, marcas comerciales, patentes y mucho más.

Derechos de autor

"https://www.netapp.com/company/legal/copyright/"

Marcas comerciales

NETAPP, el logotipo de NETAPP y las marcas enumeradas en la página de marcas comerciales de NetApp son marcas comerciales de NetApp, Inc. Los demás nombres de empresas y productos son marcas comerciales de sus respectivos propietarios.

"https://www.netapp.com/company/legal/trademarks/"

Estadounidenses

Puede encontrar una lista actual de las patentes propiedad de NetApp en:

https://www.netapp.com/pdf.html?item=/media/11887-patentspage.pdf

Política de privacidad

"https://www.netapp.com/company/legal/privacy-policy/"

Código abierto

Los archivos de notificación proporcionan información sobre los derechos de autor y las licencias de terceros que se utilizan en software de NetApp.

"Aviso sobre el sistema operativo SANtricity E-Series/EF-Series"