



Ponga en marcha la solución

BeeGFS on NetApp with E-Series Storage

NetApp
March 21, 2024

Tabla de contenidos

- Ponga en marcha la solución 1
 - Información general sobre la implementación 1
 - Obtenga más información sobre el inventario de Ansible 2
 - Revise las prácticas recomendadas 5
 - Ponga en marcha el hardware 8
 - Poner en marcha el software 9
 - Escalabilidad más allá de cinco elementos básicos 46
 - Se recomendaron porcentajes de sobreaprovisionamiento del pool de almacenamiento. 47
 - Elemento básico de gran capacidad 47

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 su 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 E-Series de NetApp, en particular el rol de ha 7.2 de BeeGFS (disponible en la ["E-Series BeeGFS GitHub de NetApp"](#)) que despliega 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:
        ictad22a01:
        ictad22a02:
        ictad22a03:
        ictad22a04:
        ictad22a05:
        ictad22a06:
    # Ansible group representing all file nodes:
    ha_cluster:
      children:
        meta_01: # Group representing a metadata service with ID 01.
          hosts:
            file_node_01: # This service is preferred on the first file
node.
            file_node_02: # And can failover to the second file node.
        meta_02: # Group representing a metadata service with ID 02.
          hosts:
            file_node_02: # This service is preferred on the second file
node.
            file_node_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>
  - i4b: <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:
  ictad22a01:
    eseries_storage_pool_configuration:
      - name: beegfs_m1_m2_m5_m6
        raid_level: raid1
        criteria_drive_count: 4
        owning_controller: A
        common_volume_configuration:
          segment_size_kb: 128
        volumes:
          - size: 21.25

```

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` aspecto: `ictad22h01`.

- 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` por lo general, puede tener nombres de host configurados para cada controladora como `ictad22a01-a` y `ictad22a01-b`, Pero ser referido en un inventario de Ansible como `ictad22a01`.

- 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
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

Esquema de direccionamiento IPoIB de cliente-servidor BeeGFS (dos subredes)

Para permitir que los clientes BeeGFS utilicen dos puertos InfiniBand, se necesitan dos subredes IPoIB con la mitad de los servicios de servidor BeeGFS configurados con una IP preferida en cada subred, a fin de garantizar que los clientes utilicen dos puertos InfiniBand para maximizar la redundancia y el posible rendimiento del sistema de archivos.

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 BeeGFS están siempre en `xxx.yyy.103.zzz` o `xxx.yyy.103.zzz`.
- Direcciones del intervalo `100.xxx.1.1` por `100.xxx.99.255` están reservados para clientes.

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

Específico	Puerto InfiniBand	Dirección IP o rango
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).



Debido a que cada bloque de creación incluye dos nodos de archivo BeeGFS, se requiere un mínimo de dos bloques de construcción para establecer el quórum en el clúster de conmutación por error. Si bien es posible configurar un clúster de dos nodos, existen limitaciones en esta configuración que pueden evitar que se produzca una conmutación al respaldo correcta en algunos casos. Si se requiere un clúster de dos nodos, también es posible incorporar un tercer dispositivo como tiebreaker, aunque este procedimiento no se contempla.

A menos que se indique lo contrario, los pasos siguientes son idénticos para cada bloque de creación del clúster independientemente de si se utiliza únicamente para ejecutar metadatos y servicios de almacenamiento o servicios de almacenamiento de BeeGFS.

Pasos

1. Configure cada nodo de archivo BeeGFS con cuatro adaptadores de canal host (HCA) de doble puerto PCIe 4.0 ConnectX-6 en modo InfiniBand e instálelos en 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 de hardware correcta para el bloque de creación BeeGFS (vista trasera).

□



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. Para cablear los nodos de archivos y bloques, utilice cables de cobre de conexión directa HDR de 1 m para 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. Utilice 2 m (o la longitud adecuada) InfiniBand HDR 200 GB cables de cobre de conexión directa para conectar los puertos InfiniBand restantes de cada nodo de archivo a los switches InfiniBand que se usarán para la red de almacenamiento.

Si hay switches InfiniBand redundantes en uso, conecte los puertos resaltados en verde claro en la siguiente figura a switches diferentes.

□

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 aprender a configurar la red para los nodos de archivo Lenovo SR665 validados, consulte ["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 8.4 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 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=8.4`. Para ver 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-8-for-x86_64
-highavailability-rpms --add=enabled:1
```

6. Actualice todo el firmware de HCA ConnectX-6 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.

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 se probaron en CentOS 8.4. Para obtener información sobre los pasos específicos de su distribución de Linux preferida, consulte ["Documentación de Ansible"](#).

1. Instale Python 3.9 y asegúrese de que la versión correcta de `pip` está instalado.

```
sudo dnf install python3.9 -y
sudo dnf install python39-pip
sudo dnf install sshpass
```

2. Cree enlaces simbólicos, asegurándose de que el binario Python 3.9 se utilice siempre que lo haga `python3` o `python` se llama.

```
sudo ln -sf /usr/bin/python3.9 /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 2.11.6]
  config file = None
  configured module search path = ['/root/.ansible/plugins/modules',
  '/usr/share/ansible/plugins/modules']
  ansible python module location = /usr/local/lib/python3.9/site-
  packages/ansible
  ansible collection location =
  /root/.ansible/collections:/usr/share/ansible/collections
  executable location = /usr/local/bin/ansible
  python version = 3.9.2 (default, Mar 10 2021, 17:29:56) [GCC 8.4.1
  20200928 (Red Hat 8.4.1-1)]
  jinja version = 3.0.2
  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 dnf 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:==3.0.1
```

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

- Utilice un sistema de control de origen como Bitbucket o Git para almacenar el contenido del directorio que contiene los archivos de libro de aplicaciones y inventario de Ansible.
- Cree un `.gitignore` Archivo que especifica los archivos que Git debe ignorar. Esto ayuda a evitar el almacenamiento de archivos grandes en Git.

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: 100.128.100. <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>.yaml` 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 addresses if
desired.
  - "pool 0.pool.ntp.org iburst maxsources 3"
  - "pool 1.pool.ntp.org iburst maxsources 3"
```

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

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

Pasos

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

```
### ha_cluster Ansible group inventory file.
# Place all default/common variables for BeeGFS HA cluster resources
below.
### Cluster node defaults
ansible_ssh_user: root
ansible_become_password: <PASSWORD>
eseries_ipoib_default_hook_templates:
  - 99-multihoming.j2 # This is required when configuring additional
static IPs (for example cluster IPs) when multiple IB ports are in the
same IPoIB subnet.
# 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 --state=active,exited |
grep eseries_nvme_ib.service"
```



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_mgmt_d_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.

- Configure un agente de cercado. (Para obtener información detallada, consulte ["Configurar la delimitación en un clúster de alta disponibilidad de Red Hat"](#).) La siguiente salida muestra ejemplos para configurar agentes de cercado 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/8/html/configuring\_and\_managing\_high\_availability\_clusters/assembly\_configuring-fencing-configuring-and-managing-high-availability-clusters.

```

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

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

Para habilitar el ajuste de rendimiento, especifique lo siguiente:

```

### Performance Configuration:
beegfs_ha_enable_performance_tuning: True

```

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

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

6. Para permitir una conectividad completa de 200 GB/HDR entre nodos de bloque y archivo, utilice el paquete Open Subnet Manager (OpenSM) del Mellanox Open Fabrics Enterprise Distribution (MLNX_OFED). (La bandeja de entrada `opensm` el paquete no admite la funcionalidad de virtualización necesaria.) Aunque se admite la puesta en marcha con Ansible, primero debe descargar los paquetes deseados en el nodo de control de Ansible que se utilice para ejecutar el rol BeeGFS.
 - a. Uso `curl` O la herramienta que desee, descargue los paquetes para la versión de OpenSM enumerados en la sección de requisitos tecnológicos del sitio web de Mellanox al `packages/` directorio. Por ejemplo:

```
curl -o packages/opensm-libs-5.9.0.MLNX20210617.c9f2ade-0.1.54103.x86_64.rpm https://linux.mellanox.com/public/repo/mlnx_ofed/5.4-1.0.3.0/rhel8.4/x86_64/opensm-libs-5.9.0.MLNX20210617.c9f2ade-0.1.54103.x86_64.rpm

curl -o packages/opensm-5.9.0.MLNX20210617.c9f2ade-0.1.54103.x86_64.rpm https://linux.mellanox.com/public/repo/mlnx_ofed/5.4-1.0.3.0/rhel8.4/x86_64/opensm-5.9.0.MLNX20210617.c9f2ade-0.1.54103.x86_64.rpm
```

- b. Rellene los siguientes parámetros en `group_vars/ha_cluster.yml` (ajuste los paquetes según sea necesario):

```

### 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-5.9.0.MLNX20210617.c9f2ade-
0.1.54103.x86_64.rpm": "/tmp/"
        "packages/opensm-5.9.0.MLNX20210617.c9f2ade-
0.1.54103.x86_64.rpm": "/tmp/"
    - packages:
      add:
        - /tmp/opensm-5.9.0.MLNX20210617.c9f2ade-
0.1.54103.x86_64.rpm
        - /tmp/opensm-libs-5.9.0.MLNX20210617.c9f2ade-
0.1.54103.x86_64.rpm
  uninstall:
    - packages:
      remove:
        - opensm
        - opensm-libs
      files:
        remove:
          - /tmp/opensm-5.9.0.MLNX20210617.c9f2ade-
0.1.54103.x86_64.rpm
          - /tmp/opensm-libs-5.9.0.MLNX20210617.c9f2ade-
0.1.54103.x86_64.rpm
eseries_ib_opensm_options:
  virt_enabled: "2"

```

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 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

# Note: At this time no other x86 servers have been qualified.
Configuration for future qualified file nodes will be added here.

```

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_11.70.2_6000_61b1131d.dlp"
eseries_firmware_nvram: "packages/N6000-872834-D06.dlp"
```

3. Descargue e instale el firmware de la unidad más reciente disponible para las unidades instaladas en los nodos de bloques desde el ["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:

```
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:
        ictad22a01:
        ictad22a02:
        ictad22a03:
        ictad22a04:
        ictad22a05:
        ictad22a06:
    # 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:`

```
    # ictad22h01/ictad22h02 HA Pair (mgmt/meta/storage building
    block):
      mgmt:
        hosts:
          ictad22h01:
          ictad22h02:
      meta_01:
        hosts:
          ictad22h01:
          ictad22h02:
      stor_01:
```

```
hosts:
  ictad22h01:
  ictad22h02:
meta_02:
  hosts:
    ictad22h01:
    ictad22h02:
stor_02:
  hosts:
    ictad22h01:
    ictad22h02:
meta_03:
  hosts:
    ictad22h01:
    ictad22h02:
stor_03:
  hosts:
    ictad22h01:
    ictad22h02:
meta_04:
  hosts:
    ictad22h01:
    ictad22h02:
stor_04:
  hosts:
    ictad22h01:
    ictad22h02:
meta_05:
  hosts:
    ictad22h02:
    ictad22h01:
stor_05:
  hosts:
    ictad22h02:
    ictad22h01:
meta_06:
  hosts:
    ictad22h02:
    ictad22h01:
stor_06:
  hosts:
    ictad22h02:
    ictad22h01:
meta_07:
  hosts:
    ictad22h02:
```

```

    ictad22h01:
  stor_07:
    hosts:
      ictad22h02:
      ictad22h01:
  meta_08:
    hosts:
      ictad22h02:
      ictad22h01:
  stor_08:
    hosts:
      ictad22h02:
      ictad22h01:

```

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_mgmgtd_conf_resource_group_options:
# <beegfs-mgmt.conf:key>:<beegfs-mgmt.conf:value>
floating_ips:
  - i1b: 100.127.101.0/16
  - i2b: 100.128.102.0/16
beegfs_service: management
beegfs_targets:
  ictad22a01:
    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:100.127.1 01.1/16 i2b:100.128.1 02.1/16	0	ictad22a01	beegfs_m1_ m2_m5_m6	A.
meta_02.yml	8025	i2b: 100.128.102. 2/16 i1b:100.127.1 01.2/16	0	ictad22a01	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:100.127.1 01.3/16 i4b:100.128.1 02.3/16	1	ictad22a02	beegfs_m3_ m4_m7_m8	A.
meta_04.yml	8045	i4b:100.128.1 02.4/16 i3b:100.127.1 01.4/16	1	ictad22a02	beegfs_m3_ m4_m7_m8	B
meta_05.yml	8055	i1b:100.127.1 01.5/16 i2b:100.128.1 02.5/16	0	ictad22a01	beegfs_m1_ m2_m5_m6	A.
meta_06.yml	8065	i2b: 100.128.102. 6/16 i1b:100.127.1 01.6/16	0	ictad22a01	beegfs_m1_ m2_m5_m6	B
meta_07.yml	8075	i3b:100.127.1 01.7/16 i4b:100.128.1 02.7/16	1	ictad22a02	beegfs_m3_ m4_m7_m8	A.
meta_08.yml	8085	i4b:100.128.1 02.8/16 i3b:100.127.1 01.8/16	1	ictad22a02	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 sobreprovisionamiento 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:100.127.1 03.1/16 i2b:100.128.1 04.1/16	0	ictad22a01	beegfs_s1_s2	A.
stor_02.yml	8023	i2b: 100.128.104. 2/16 i1b:100.127.1 03.2/16	0	ictad22a01	beegfs_s1_s2	B
stor_03.yml	8033	i3b:100.127.1 03.3/16 i4b:100.128.1 04.3/16	1	ictad22a02	beegfs_s3_s4	A.
stor_04.yml	8043	i4b:100.128.1 04.4/16 i3b:100.127.1 03.4/16	1	ictad22a02	beegfs_s3_s4	B

Nombre de archivo	Puerto	IP flotantes	Zona NUMA	Nodo de bloques	Del banco de almacenamiento	Controladora propietaria
stor_05.yml	8053	i1b:100.127.1 03.5/16 i2b:100.128.1 04.5/16	0	ictad22a01	beegfs_s5_s6	A.
stor_06.yml	8063	i2b: 100.128.104. 6/16 i1b:100.127.1 03.6/16	0	ictad22a01	beegfs_s5_s6	B
stor_07.yml	8073	i3b:100.127.1 03.7/16 i4b:100.128.1 04.7/16	1	ictad22a02	beegfs_s7_s8	A.
stor_08.yml	8083	i4b:100.128.1 04.8/16 i3b:100.127.1 03.8/16	1	ictad22a02	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. Pulga `inventory.yml`, rellene los siguientes parámetros bajo la configuración existente:

```

meta_09:
  hosts:
    ictad22h03:
    ictad22h04:
stor_09:
  hosts:
    ictad22h03:
    ictad22h04:
meta_10:
  hosts:
    ictad22h03:
    ictad22h04:
stor_10:
  hosts:
    ictad22h03:
    ictad22h04:
meta_11:
  hosts:

```

```
    ictad22h03:
    ictad22h04:
stor_11:
  hosts:
    ictad22h03:
    ictad22h04:
meta_12:
  hosts:
    ictad22h03:
    ictad22h04:
stor_12:
  hosts:
    ictad22h03:
    ictad22h04:
meta_13:
  hosts:
    ictad22h04:
    ictad22h03:
stor_13:
  hosts:
    ictad22h04:
    ictad22h03:
meta_14:
  hosts:
    ictad22h04:
    ictad22h03:
stor_14:
  hosts:
    ictad22h04:
    ictad22h03:
meta_15:
  hosts:
    ictad22h04:
    ictad22h03:
stor_15:
  hosts:
    ictad22h04:
    ictad22h03:
meta_16:
  hosts:
    ictad22h04:
    ictad22h03:
stor_16:
  hosts:
    ictad22h04:
    ictad22h03:
```

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 sobreaaprovisionamiento del pool de almacenamiento](#)".

Nombre de archivo	Puerto	IP flotantes	Zona NUMA	Nodo de bloques	Del banco de almacenamiento	Controladora propietaria
<code>meta_09.yml</code>	8015	i1b:100.127.1 01.9/16 i2b:100.128.1 02.9/16	0	ictad22a03	beegfs_m9_ m10_m13_m 14	A.
<code>meta_10.yml</code>	8025	i2b: 100.128.102. 10/16 i1b:100.127.1 01.10/16	0	ictad22a03	beegfs_m9_ m10_m13_m 14	B
<code>meta_11.yml</code>	8035	i3b:100.127.1 01.11/16 i4b:100.128.1 02.11/16	1	ictad22a04	beegfs_m11_ m12_m15_m 16	A.

Nombre de archivo	Puerto	IP flotantes	Zona NUMA	Nodo de bloques	Del banco de almacenamiento	Controladora propietaria
meta_12.yml	8045	i4b:100.128.1 02.12/16 i3b:100.127.1 01.12/16	1	ictad22a04	beegfs_m11_ m12_m15_m 16	B
meta_13.yml	8055	i1b:100.127.1 01.13/16 i2b:100.128.1 02.13/16	0	ictad22a03	beegfs_m9_ m10_m13_m 14	A.
meta_14.yml	8065	i2b: 100.128.102. 14/16 i1b:100.127.1 01.14/16	0	ictad22a03	beegfs_m9_ m10_m13_m 14	B
meta_15.yml	8075	i3b:100.127.1 01.15/16 i4b:100.128.1 02.15/16	1	ictad22a04	beegfs_m11_ m12_m15_m 16	A.
meta_16.yml	8085	i4b:100.128.1 02.16/16 i3b:100.127.1 01.16/16	1	ictad22a04	beegfs_m11_ m12_m15_m 16	B

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

```

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

```



Para ver el tamaño correcto de uso, consulte "[Se recomendaron porcentajes de sobreprovisionamiento 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:100.127.1 03.9/16 i2b:100.128.1 04.9/16	0	ictad22a03	beegfs_s9_s1 0	A.
stor_10.yml	8023	i2b: 100.128.104. 10/16 i1b:100.127.1 03.10/16	0	ictad22a03	beegfs_s9_s1 0	B
stor_11.yml	8033	i3b:100.127.1 03.11/16 i4b:100.128.1 04.11/16	1	ictad22a04	beegfs_s11_s 12	A.
stor_12.yml	8043	i4b:100.128.1 04.12/16 i3b:100.127.1 03.12/16	1	ictad22a04	beegfs_s11_s 12	B

Nombre de archivo	Puerto	IP flotantes	Zona NUMA	Nodo de bloques	Del banco de almacenamiento	Controladora propietaria
stor_13.yml	8053	i1b:100.127.1 03.13/16 i2b:100.128.1 04.13/16	0	ictad22a03	beegfs_s13_s 14	A.
stor_14.yml	8063	i2b: 100.128.104. 14/16 i1b:100.127.1 03.14/16	0	ictad22a03	beegfs_s13_s 14	B
stor_15.yml	8073	i3b:100.127.1 03.15/16 i4b:100.128.1 04.15/16	1	ictad22a04	beegfs_s15_s 16	A.
stor_16.yml	8083	i4b:100.128.1 04.16/16 i3b:100.127.1 03.16/16	1	ictad22a04	beegfs_s15_s 16	B

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

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

Pasos

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

```

# ictad22h05/ictad22h06 HA Pair (storage only building block):
stor_17:
  hosts:
    ictad22h05:
    ictad22h06:
stor_18:
  hosts:
    ictad22h05:
    ictad22h06:
stor_19:
  hosts:
    ictad22h05:
    ictad22h06:
stor_20:
  hosts:
    ictad22h05:
    ictad22h06:
stor_21:
  hosts:
    ictad22h06:
    ictad22h05:
stor_22:
  hosts:
    ictad22h06:
    ictad22h05:
stor_23:
  hosts:
    ictad22h06:
    ictad22h05:
stor_24:
  hosts:
    ictad22h06:
    ictad22h05:

```

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 almacenamie nto	Controladora propietaria
stor_17.yml	8013	i1b:100.127.1 03.17/16 i2b:100.128.1 04.17/16	0	ictad22a05	beegfs_s17_s 18	A.
stor_18.yml	8023	i2b: 100.128.104. 18/16 i1b:100.127.1 03.18/16	0	ictad22a05	beegfs_s17_s 18	B
stor_19.yml	8033	i3b:100.127.1 03.19/16 i4b:100.128.1 04.19/16	1	ictad22a06	beegfs_s19_s 20	A.

Nombre de archivo	Puerto	IP flotantes	Zona NUMA	Nodo de bloques	Del banco de almacenamiento	Controladora propietaria
stor_20.yml	8043	i4b:100.128.104.20/16 i3b:100.127.103.20/16	1	ictad22a06	beegfs_s19_s20	B
stor_21.yml	8053	i1b:100.127.103.21/16 i2b:100.128.104.21/16	0	ictad22a05	beegfs_s21_s22	A.
stor_22.yml	8063	i2b:100.128.104.22/16 i1b:100.127.103.22/16	0	ictad22a05	beegfs_s21_s22	B
stor_23.yml	8073	i3b:100.127.103.23/16 i4b:100.128.104.23/16	1	ictad22a06	beegfs_s23_s24	A.
stor_24.yml	8083	i4b:100.128.104.24/16 i3b:100.127.103.24/16	1	ictad22a06	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.
        import_role:
          name: beegfs_ha_7_2

```



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.

- 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``, 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 puestas en marcha más grandes, sobreponiendo el número predeterminado de horquillas (5) utilizando el `forks` Se recomienda aumentar el número de hosts que Ansible configura en paralelo. (Para obtener más información, consulte "[Ajuste del rendimiento de Ansible](#)" y.. "[Control de la ejecución del libro de estrategia](#)".) El valor máximo depende de la potencia de procesamiento disponible en el nodo de control 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>0
    address: <IP>/<SUBNET_MASK>
```



Actualmente, deben configurarse dos interfaces InfiniBand en cada cliente, una en cada una de las dos subredes IPoIB de almacenamiento. Si se utilizan subredes de ejemplo y rangos recomendados para cada servicio BeeGFS que se indica aquí, los clientes deben tener una interfaz configurada en el intervalo de 100.127.1. 0 para 100.127.99.255 y el otro en 100.128.1. 0 para 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:
      ictad21h01:
      ictad21h02:
      ictad21h03:
      ictad21h04:
      ictad21h05:
      ictad21h06:
      ictad21h07:
      ictad21h08:
      ictad21h09:
      ictad21h10:
    vars:
      # OPTION 1: If you're using the Mellanox 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 Mellanox OFED, asegúrese de que `beegfs_client_ofed_include_path` Apunta a la "ruta de inclusión de encabezado" correcta 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:
      - sysMgmtHost: 100.127.101.0 # Primary IP of the BeeGFS
management service.
      mount_point: /mnt/beegfs      # Path to mount BeeGFS on the
client.
      connInterfaces:
        - <INTERFACE> # Example: ibs4f1
        - <INTERFACE>
      beegfs_client_config:
        # Maximum number of simultaneous connections to the same
node.

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

```



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_mgmt_d_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.



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

Se recomendaron porcentajes de sobreaprovisionamiento del pool de almacenamiento

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

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

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



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

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

Elemento básico de gran capacidad

La guía de implementación de la solución BeeGFS estándar describe procedimientos y

recomendaciones para requisitos de alto rendimiento en las cargas de trabajo. Los clientes que busquen cumplir requisitos de alta capacidad deben observar las variaciones en la implementación y las recomendaciones que se describen aquí.

□

Controladoras

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

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

Ubicación de la unidad

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

□

Bandejas de expansión

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

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

Información de copyright

Copyright © 2024 NetApp, Inc. Todos los derechos reservados. Imprimido en EE. UU. No se puede reproducir este documento protegido por copyright ni parte del mismo de ninguna forma ni por ningún medio (gráfico, electrónico o mecánico, incluidas fotocopias, grabaciones o almacenamiento en un sistema de recuperación electrónico) sin la autorización previa y por escrito del propietario del copyright.

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

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

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

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

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

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

Información de la marca comercial

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