



## **Definieren Sie das BeeGFS-Dateisystem**

BeeGFS on NetApp with E-Series Storage

NetApp  
January 27, 2026

# Inhalt

Definieren Sie das BeeGFS-Dateisystem . . . . .	1
Ansible-Bestandsübersicht . . . . .	1
Überblick . . . . .	1
Schritte . . . . .	1
Planen Sie das Dateisystem . . . . .	2
Überblick . . . . .	2
Schritte . . . . .	2
Datei- und Blockknoten definieren . . . . .	3
Konfigurieren Einzelner Dateiknoten . . . . .	3
Konfigurieren Einzelner Blockknoten . . . . .	7
Festlegen Der Konfiguration Des Gemeinsamen Dateiknotens . . . . .	11
Festlegen Der Konfiguration Allgemeiner Blockknoten . . . . .	18
BeeGFS-Dienste definieren . . . . .	21
Definieren Sie den BeeGFS-Managementdienst . . . . .	21
Definieren Sie den BeeGFS-Metadatendienst . . . . .	23
Definieren Sie den BeeGFS-Speicherdienst . . . . .	25
Zuordnen von BeeGFS-Services zu Datei-Nodes . . . . .	27
Überblick . . . . .	27
Schritte . . . . .	27

# Definieren Sie das BeeGFS-Dateisystem

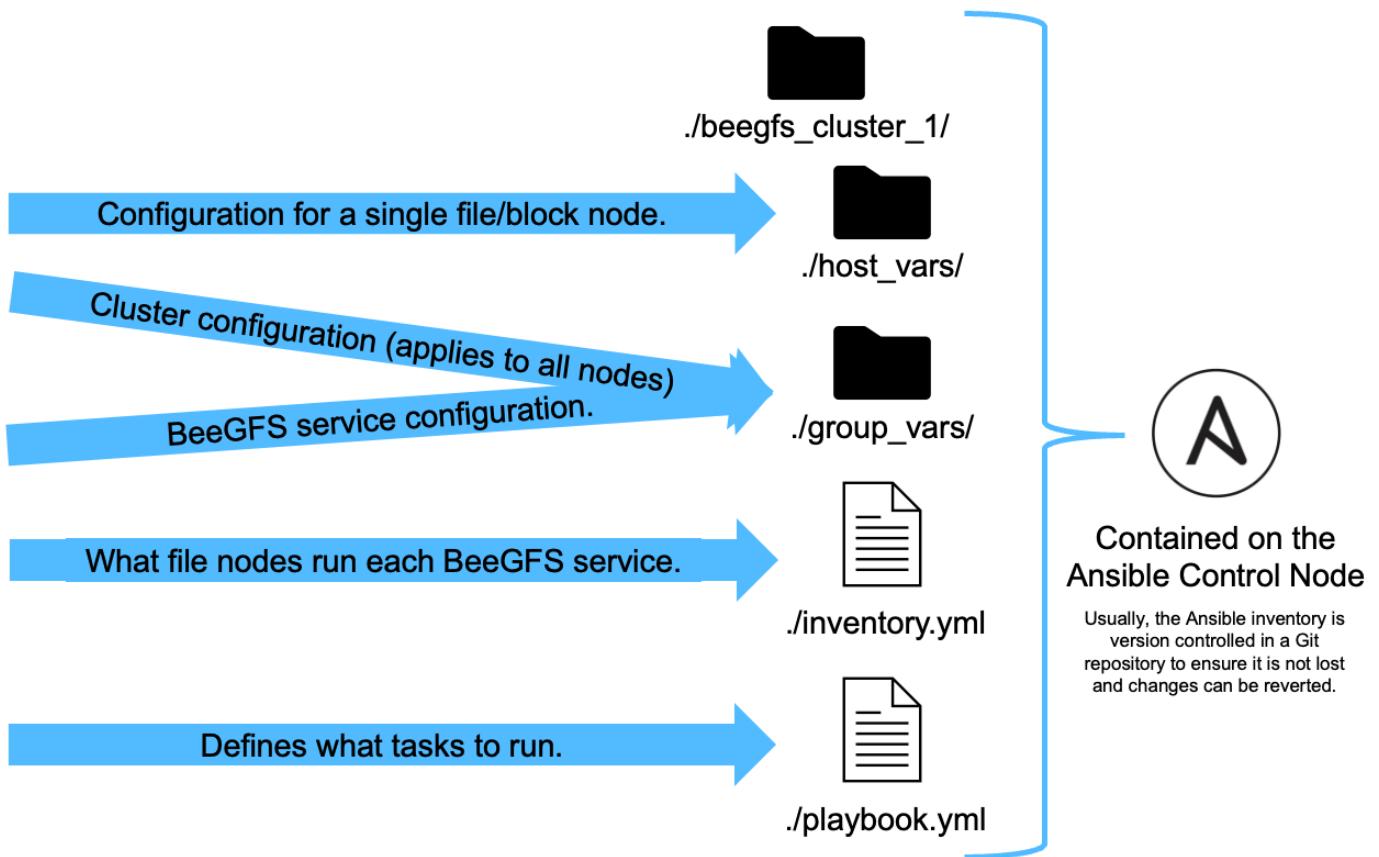
## Ansible-Bestandsübersicht

Der Ansible-Bestand ist ein Satz von Konfigurationsdateien, die das gewünschte BeeGFS HA-Cluster definieren.

### Überblick

Es wird empfohlen, die standardmäßigen Ansible-Methoden für die Organisation des zu befolgen "Inventar", Einschließlich der Verwendung von "Unterverzeichnisse/Dateien" Anstatt den gesamten Bestand in einer Datei zu speichern.

Der Ansible-Bestand für ein einzelnes BeeGFS HA-Cluster ist wie folgt organisiert:



**i** Da ein einzelnes BeeGFS-Dateisystem mehrere HA-Cluster umfassen kann, können große Installationen mehrere Ansible-Inventare durchführen. Im Allgemeinen wird es nicht empfohlen, mehrere HA-Cluster als einen einzelnen Ansible-Bestand zu definieren, um Probleme zu vermeiden.

## Schritte

1. Erstellen Sie auf Ihrem Ansible-Kontroll-Node ein leeres Verzeichnis, das den Ansible-Bestand für das BeeGFS-Cluster enthält, das bereitgestellt werden soll.

- a. Wenn Ihr Filesystem schließlich mehrere HA-Cluster enthalten soll/kann, wird empfohlen, zuerst ein Verzeichnis für das Dateisystem zu erstellen und dann Unterverzeichnisse für den Bestand, der die einzelnen HA-Cluster darstellt, einzutragen. Beispiel:

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

2. Erstellen Sie im Verzeichnis, das den Bestand für den HA-Cluster enthält, den Sie bereitstellen möchten, zwei Verzeichnisse `group_vars` Und `host_vars` Und zwei Dateien `inventory.yml` Und `playbook.yml`.

Die folgenden Abschnitte gehen durch die Definition des Inhalts jeder dieser Dateien.

## Planen Sie das Dateisystem

Planen Sie die Filesystem-Implementierung, bevor Sie den Ansible-Bestand aufbauen.

### Überblick

Vor der Bereitstellung des Dateisystems sollten Sie festlegen, welche IP-Adressen, Ports und andere Konfigurationen für alle Datei-Nodes, Block-Nodes und BeeGFS-Services im Cluster erforderlich sind. Während die genaue Konfiguration je nach Architektur des Clusters variiert, werden in diesem Abschnitt Best Practices und Schritte definiert, die allgemein anwendbar sind.

### Schritte

- Wenn Sie zum Verbinden von Datei-Nodes mit Block-Nodes ein IP-basiertes Storage-Protokoll (z. B. iSER, iSCSI, NVMe/IB oder NVMe/RoCE) verwenden, füllen Sie für jeden Baustein das folgende Arbeitsblatt aus. Jede direkte Verbindung in einem einzelnen Baustein sollte ein eigenes Subnetz haben, und es sollte keine Überschneidung mit Subnetzen geben, die für die Client-Server-Konnektivität verwendet werden.

Datei-Node	IB-Port	IP-Adresse	Block-Node	IB-Port	Physische IP-Adresse	Virtuelle IP (nur für EF600 mit HDR IB)
<HOSTNAME>	<PORT>	<IP/SUBNET>	<HOSTNAME>	<PORT>	<IP/SUBNET>	<IP/SUBNET>



Wenn Datei- und Block-Nodes in jedem Baustein direkt verbunden sind, können für mehrere Bausteine oft dieselben IPs/Schemas verwendet werden.

- Füllen Sie unabhängig davon aus, ob Sie InfiniBand oder RDMA over Converged Ethernet (RoCE) für das Storage-Netzwerk verwenden, das folgende Arbeitsblatt aus, um die IP-Bereiche zu ermitteln, die für HA-Cluster-Services, BeeGFS-Fileservices und Clients verwendet werden, um zu kommunizieren:

Zweck	InfiniBand-Port	IP-Adresse oder Bereich
BeeGFS Cluster-IP(s)	<INTERFACE(s)>	<RANGE>
BeeGFS Management	<INTERFACE(s)>	<IP(s)>
BeeGFS-Metadaten	<INTERFACE(s)>	<RANGE>
BeeGFS-Speicherung	<INTERFACE(s)>	<RANGE>
BeeGFS-Clients	<INTERFACE(s)>	<RANGE>

- a. Wenn Sie ein einzelnes IP-Subnetz verwenden, ist nur ein Arbeitsblatt erforderlich. Füllen Sie andernfalls auch ein Arbeitsblatt für das zweite Subnetz aus.
3. Füllen Sie für jeden Baustein im Cluster das folgende Arbeitsblatt aus, um festzulegen, welche BeeGFS-Services ausgeführt werden sollen. Geben Sie für jeden Service die bevorzugten/sekundären Dateiknoten, Netzwerkport, fließende IP(s), NUMA-Zonenzuweisung (falls erforderlich) und welche Block-Nodes für ihre Ziele verwendet werden. Beachten Sie beim Ausfüllen des Arbeitsblatts die folgenden Richtlinien:
- Geben Sie BeeGFS-Dienste als entweder an mgmt.yml, meta\_<ID>.yml, Oder storage\_<ID>.yml Wobei ID eine eindeutige Nummer für alle BeeGFS-Dienste dieses Typs in diesem Dateisystem darstellt. Dieses Übereinkommen erleichtert die Rückverweisen auf dieses Arbeitsblatt in den nachfolgenden Abschnitten, während Dateien für die Konfiguration der einzelnen Dienste erstellt werden.
  - Ports für BeeGFS-Dienste müssen nur in einem bestimmten Baustein einzigartig sein. Stellen Sie sicher, dass Dienste mit derselben Portnummer nicht jemals auf demselben Dateiknoten ausgeführt werden können, um Portkonflikte zu vermeiden.
  - Bei Bedarf können Services Volumes von mehr als einem Block-Node und/oder Storage-Pool nutzen (und nicht alle Volumes müssen Eigentum desselben Controllers sein). Zudem können mehrere Services denselben Block-Node und/oder dieselbe Storage-Pool-Konfiguration nutzen (einzelne Volumes werden in einem späteren Abschnitt definiert).

BeeGFS-Dienst (Dateiname)	File-Nodes	Port	Fließende IPs	NUMA-Zone	Block-Node	Storage-Pool	Controller, der die LUN besitzt
<SERVICE TYPE>_<ID>.yml	<PREFER RED 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>

Weitere Details zu Standardkonventionen, Best Practices und ausgefüllten Arbeitsblättern finden Sie in den "[Best Practices in sich vereint](#)" "Definieren Sie BeeGFS-Bausteine" Abschnitten und der BeeGFS on NetApp Verified Architecture.

## Datei- und Blockknoten definieren

### Konfigurieren Einzelner Dateiknoten

Legen Sie die Konfiguration für einzelne Datei-Nodes mithilfe von Host-Variablen fest

## (Host\_vars).

### Überblick

In diesem Abschnitt wird das Ausfüllen von erläutert `host_vars/<FILE_NODE_HOSTNAME>.yml` Datei für jeden Datei-Node im Cluster. Diese Dateien sollten nur die für einen bestimmten Dateiknoten spezifische Konfiguration enthalten. Hierzu zählen folgende allgemein:

- Die Definition der IP oder des Hostnamen Ansible sollte für die Verbindung mit dem Node verwendet werden.
- Konfiguration zusätzlicher Schnittstellen und Cluster-IPs, die für HA-Cluster-Services (Pacemaker und Corosync) zur Kommunikation mit anderen Datei-Nodes verwendet werden Standardmäßig verwenden diese Dienste dasselbe Netzwerk wie die Managementoberfläche, aber für Redundanz sollten zusätzliche Schnittstellen verfügbar sein. Es ist üblich, zusätzliche IPs im Storage-Netzwerk zu definieren, ohne dass ein zusätzliches Cluster- oder Management-Netzwerk erforderlich ist.
  - Die Performance aller Netzwerke, die für die Cluster-Kommunikation verwendet werden, ist für die Performance des Filesystems nicht von entscheidender Bedeutung. Bei der Standardkonfiguration des Clusters bietet ein Netzwerk mit mindestens 1 GB/s im Allgemeinen ausreichende Performance für Cluster-Vorgänge, z. B. die Synchronisierung von Node-Status und die Koordinierung von Änderungen des Clusterressourcenstatus. Langsame/überlastete Netzwerke können dazu führen, dass Änderungen des Ressourcenzustands länger dauern als üblich, und in extremen Fällen können Nodes aus dem Cluster entfernt werden, wenn sie in einem angemessenen Zeitrahmen keine Herzschläge senden können.
- Konfigurieren von Schnittstellen, die für die Verbindung zu Block-Nodes über das gewünschte Protokoll verwendet werden (z. B. iSCSI/iSER, NVMe/IB, NVMe/RoCE, FCP usw.)

### Schritte

Wenn Sie auf das im "[Planen Sie das Dateisystem](#)" Abschnitt definierte IP-Adressierungsschema verweisen, erstellen Sie für jeden Dateiknoten im Cluster eine Datei `host_vars/<FILE_NODE_HOSTNAME>/yml` und füllen Sie sie wie folgt aus:

1. Geben Sie oben die IP oder den Hostnamen an, den Ansible für den Node mit SSH verwenden und verwalten soll:

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

2. Konfigurieren Sie zusätzliche IPs, die für den Cluster-Datenverkehr verwendet werden können:

- a. Wenn der Netzwerktyp ist "[InfiniBand \(mit IPoIB\)](#)":

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

- b. Wenn der Netzwerktyp ist "[RDMA über Converged Ethernet \(RoCE\)](#)":

```

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. Wenn der Netzwerktyp ist "Ethernet (nur TCP, kein 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. Geben Sie an, welche IPs für Cluster-Datenverkehr verwendet werden sollen, wobei die bevorzugten IPs höher aufgeführt sind:

```

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.

```



IPS, die in Schritt zwei konfiguriert sind, werden nicht als Cluster-IPs verwendet, es sei denn, sie sind im enthalten beegfs\_ha\_cluster\_node\_ips Liste. So können mithilfe von Ansible zusätzliche IPs/Schnittstellen konfiguriert werden, die bei Bedarf für andere Zwecke verwendet werden können.

4. Wenn der Datei-Node über ein IP-basiertes Protokoll mit Block-Nodes kommunizieren muss, müssen IPs auf der entsprechenden Schnittstelle konfiguriert werden. Für dieses installierte/konfigurierte Protokoll sind alle Pakete erforderlich.

a. Bei Verwendung von "ISCSI":

```

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

```

b. Bei Verwendung von "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. Bei Verwendung von "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. Bei Verwendung von "NVMe/RoCE":

```

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

```

e. Andere Protokolle:

- Bei Verwendung von "NVMe/FC", Die Konfiguration einzelner Schnittstellen ist nicht erforderlich. Die BeeGFS-Cluster-Implementierung erkennt das Protokoll automatisch und installiert/konfiguriert die Anforderungen nach Bedarf. Wenn Sie eine Fabric zum Verbinden von Datei- und Block-Nodes verwenden, stellen Sie sicher, dass die Switches im Rahmen der Best Practices von NetApp und dem Switch-Anbieter ordnungsgemäß begrenzt sind.
- Bei der Verwendung von FCP oder SAS muss keine zusätzliche Software installiert oder konfiguriert werden. Wenn Sie FCP verwenden, stellen Sie sicher, dass die Switches Folgendes ordnungsgemäß begrenzt sind "NetApp" Und die Best Practices Ihres Switch-Anbieters berücksichtigen.
- Die Verwendung von IB-SRP wird derzeit nicht empfohlen. NVMe/IB oder iSER nutzen, je nachdem, was die Block-Nodes der E-Series unterstützen.

Klicken Sie Auf "Hier" Beispiel für eine komplette Bestandsdatei, die einen einzelnen Dateiknoten darstellt.

**Erweitert: Zwischen Ethernet- und InfiniBand-Modus können NVIDIA ConnectX VPI-Adapter eingesetzt werden**

NVIDIA ConnectX-Virtual Protocol Interconnect&reg; (VPI) Adapter unterstützen sowohl InfiniBand als auch Ethernet als Transportebene. Das Umschalten zwischen den Modi wird nicht automatisch ausgehandelt und muss mit dem in enthaltenen Werkzeug konfiguriert werden <code>mstconfig</code> <code>mstflint</code>, einem Open-Source-Paket, das Teil des ist <a href="https://docs.nvidia.com/networking/display/mftv4270/mft+supported+configurations+and+parameters" target="\_blank">"NVIDIA Firmware Tools (MFT)"</a>. Das Ändern des Modus der Adapter muss nur einmal vorgenommen werden. Dies kann manuell vorgenommen oder als Teil aller Schnittstellen, die mithilfe des Abschnitts des Bestands konfiguriert wurden, in das Ansible-Inventar aufgenommen <code>eseries-[ib|ib\_iser|ipoib|nvme\_ib|nvme\_roce|roce]\_interfaces:</code> werden, um es automatisch prüfen/anwenden zu

lassen.

So kann beispielsweise die aktuelle Schnittstellenspannung im InfiniBand-Modus in Ethernet geändert werden, damit sie für RoCE verwendet werden kann:

1. Für jede Schnittstelle, die Sie angeben möchten `mstconfig` Als Zuordnung (oder Wörterbuch), das angibt `LINK_TYPE_P<N>` Wo <N> Wird durch die Anschlussnummer des HCA für die Schnittstelle bestimmt. Der <N> Wert kann durch Ausführen bestimmt werden `grep PCI_SLOT_NAME /sys/class/net/<INTERFACE_NAME>/device/uevent` Und fügen Sie 1 zur letzten Nummer aus dem PCI-Steckplatznamen hinzu und konvertieren Sie auf dezimal.
  - a. Beispiel angegeben `PCI_SLOT_NAME=0000:2f:00.2` ( $2 + 1 \rightarrow$  HCA-Port 3) → `LINK_TYPE_P3`:  
`eth`:

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

Weitere Details finden Sie im "[Dokumentation der NetApp E-Series Host-Sammlung](#)" Für den Schnittstellentyp/das Protokoll, das Sie verwenden.

## Konfigurieren Einzelner Blockknoten

Legen Sie die Konfiguration für einzelne Block-Nodes mithilfe von Host-Variablen fest (Host\_vars).

### Überblick

In diesem Abschnitt wird das Ausfüllen von erläutert `host_vars/<BLOCK_NODE_HOSTNAME>.yml` Datei für jeden Block-Node im Cluster. Diese Dateien sollten nur die Konfiguration enthalten, die für einen bestimmten Block-Node eindeutig ist. Hierzu zählen folgende allgemein:

- Der Systemname (wie in System Manager angezeigt).
- Die HTTPS-URL für einen der Controller (wird zum Verwalten des Systems mit seiner REST-API verwendet).
- Welche Storage-Protokoll-Datei-Nodes verwenden für die Verbindung zu diesem Block-Node?
- Konfigurieren von Ports für die Host-Schnittstelle (HIC), z. B. IP-Adressen (falls erforderlich)

### Schritte

Wenn Sie auf das im "[Planen Sie das Dateisystem](#)" Abschnitt definierte IP-Adressierungsschema verweisen, erstellen Sie für jeden Block-Node im Cluster eine Datei `host_vars/<BLOCK_NODE_HOSTNAME>/yml` und füllen Sie sie wie folgt aus:

1. Geben Sie oben den Systemnamen und die HTTPS-URL für einen Controller an:

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

2. Wählen Sie die aus "Protokoll" Dateiknoten werden für die Verbindung zu diesem Block-Knoten verwendet:

a. Unterstützte Protokolle: auto, iscsi, fc, sas, ib\_srp, ib\_iser, nvme\_ib, nvme\_fc, nvme\_roce.

```
eseries_initiator_protocol: <PROTOCOL>
```

3. Je nach verwendetem Protokoll erfordern die HIC-Ports unter Umständen zusätzliche Konfigurationen. Bei Bedarf sollte die HIC-Port-Konfiguration definiert werden, sodass der oberste Eintrag in der Konfiguration für jeden Controller dem physischen, am meisten linken Port auf jedem Controller entspricht, und der untere Port dem fast rechten Port. Alle Ports erfordern eine gültige Konfiguration, auch wenn sie derzeit nicht verwendet werden.



Wenn Sie HDR (200 GB) InfiniBand oder 200 GB RoCE mit EF600 Block-Nodes verwenden, sehen Sie den folgenden Abschnitt.

a. Für 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. Für 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. Für NVMe/IB:

```

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

```

d. Für 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. FC- und SAS-Protokolle erfordern keine zusätzliche Konfiguration. SRP wird nicht richtig empfohlen.

Weitere Optionen zum Konfigurieren von HIC-Ports und Hostprotokollen, einschließlich der Möglichkeit zum Konfigurieren von iSCSI-CHAP finden Sie im "[Dokumentation](#)" In der SANtricity Kollektion enthalten. Hinweis: Bei der Bereitstellung von BeeGFS werden der Speicherpool, die Volume-Konfiguration und andere Aspekte des Bereitstellungsspeicher an anderer Stelle konfiguriert und in dieser Datei nicht definiert.

Klicken Sie Auf "[Hier](#)" Beispiel für eine komplette Bestandsdatei, die einen einzelnen Block-Knoten darstellt.

#### Mit HDR (200 GB) InfiniBand oder 200 GB RoCE mit NetApp EF600 Block-Nodes:

Um HDR (200 GB) InfiniBand mit der EF600 zu verwenden, muss für jeden physischen Port eine zweite „virtuelle“ IP konfiguriert werden. Nachfolgend sehen Sie ein Beispiel für die korrekte Konfiguration eines EF600 mit Dual-Port InfiniBand HDR HIC:

```
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)
```

## Festlegen Der Konfiguration Des Gemeinsamen Dateiknotens

Geben Sie unter Verwendung von Gruppenvariablen (Group\_vars) die Konfiguration allgemeiner Dateiknoten an.

### Überblick

Konfiguration, die auf alle Datei-Nodes Apfel soll, wird bei definiert group\_vars/ha\_cluster.yml. Dazu gehören in der Regel:

- Details zur Verbindung und Anmeldung zu den einzelnen Dateiknoten.
- Gängige Netzwerkkonfiguration.
- Gibt an, ob ein automatischer Neustart zulässig ist.
- Wie Firewall- und selinux-Status konfiguriert werden sollen.
- Cluster-Konfiguration mit Warn- und Fechten
- Performance-Optimierung:
- Allgemeine BeeGFS-Servicekonfiguration.



Die in dieser Datei festgelegten Optionen können auch auf einzelnen Datei-Nodes definiert werden, z. B. wenn gemischte Hardware-Modelle verwendet werden oder Sie unterschiedliche Passwörter für jeden Knoten haben. Die Konfiguration auf einzelnen Datei-Knoten hat Vorrang vor der Konfiguration in dieser Datei.

## Schritte

Erstellen Sie die Datei `group_vars/ha_cluster.yml` und füllen Sie es wie folgt aus:

1. Geben Sie an, wie sich der Ansible Control-Node mit den Remote-Hosts authentifizieren soll:

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



Speichern Sie Passwörter insbesondere für Produktionsumgebungen nicht im Klartext. Verwenden Sie stattdessen Ansible Vault (siehe "["Verschlüsseln von Inhalten mit Ansible Vault"](#)) Oder der `--ask-become-pass` Option beim Ausführen des Playbooks. Wenn der `ansible_ssh_user` ist bereits root, dann können Sie optional auslassen `ansible_become_password`.

2. Wenn Sie statische IPs in ethernet- oder InfiniBand-Schnittstellen konfigurieren (zum Beispiel Cluster-IPs) und mehrere Schnittstellen im gleichen IP-Subnetz sind (z. B. wenn ib0 192.168.1.10/24 verwendet und ib1 192.168.1.11/24 verwendet), Zusätzliche IP-Routing-Tabellen und -Regeln müssen so eingerichtet sein, dass Multi-Homed-Unterstützung ordnungsgemäß funktioniert. Aktivieren Sie einfach den mitgelieferten Konfigurationshaken für Netzwerkschnittstellen wie folgt:

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

3. Bei der Implementierung des Clusters müssen je nach Storage-Protokoll Nodes neu gestartet werden, um die Erkennung von Remote-Block-Geräten (E-Series Volumes) zu erleichtern oder andere Aspekte der Konfiguration anzuwenden. Standardmäßig werden vor dem Neubooten von Nodes angezeigt. Sie können Nodes jedoch durch Angabe des folgenden Verfahrens automatisch neu starten:

```
eseries_common_allow_host_reboot: true
```

- a. Standardmäßig nach einem Neustart, um sicherzustellen, dass Block-Geräte und andere Services bereit sind Ansible wartet, bis das System `default.target` erreicht wird, bevor die Implementierung fortgesetzt wird. In manchen Szenarien, in denen NVMe/IB verwendet wird, ist dies möglicherweise nicht lang genug, um Remote-Geräte zu initialisieren, zu erkennen und eine Verbindung zu herstellen. Dies kann dazu führen, dass die automatisierte Implementierung vorzeitig ausfällt und ausfällt. Um dies bei der Nutzung von NVMe/IB zu vermeiden, müssen Sie außerdem Folgendes definieren:

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

4. Für die Kommunikation mit BeeGFS und HA-Cluster-Services sind mehrere Firewall-Ports erforderlich. Wenn Sie die Firewall nicht manuell konfigurieren möchten (nicht empfohlen), geben Sie Folgendes an, damit erforderliche Firewall-Zonen erstellt und Ports automatisch geöffnet werden:

```
beegfs_ha_firewall_configure: True
```

5. Derzeit wird SELinux nicht unterstützt, und es wird empfohlen, den Status auf deaktiviert zu setzen, um Konflikte zu vermeiden (insbesondere, wenn RDMA verwendet wird). Stellen Sie Folgendes ein, um sicherzustellen, dass SELinux deaktiviert ist:

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

6. Konfigurieren Sie die Authentifizierung so, dass Dateiknoten kommunizieren können und passen Sie die Standardeinstellungen entsprechend Ihren Unternehmensrichtlinien an:

```
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. "Planen Sie das Dateisystem" Legen Sie auf der Grundlage des Abschnitts die BeeGFS-Management-IP für dieses Dateisystem fest:

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



Während scheinbar redundant, `beegfs_ha_mgmtd_floating_ip` ist wichtig, wenn Sie das BeeGFS-Dateisystem über einen einzelnen HA-Cluster hinaus skalieren. Nachfolgende HA-Cluster werden ohne zusätzlichen BeeGFS-Managementservice bereitgestellt und Punkt am Managementservice des ersten Clusters.

8. Aktivieren Sie bei Bedarf E-Mail-Alarme:

```

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. Es wird dringend empfohlen, Fechten zu aktivieren, da bei Ausfall des primären Knotens Services vom Starten auf sekundären Knoten blockiert werden können.

- a. Aktivieren Sie das globale Fechten, indem Sie Folgendes angeben:

```

beegfs_ha_cluster_crm_config_options:
    stonith-enabled: True

```

- i. Hinweis: Bei Bedarf können auch alle unterstützten "Cluster-Eigenschaft" Daten hier angegeben werden. Diese Anpassungen sind in der Regel nicht notwendig, da die BeeGFS HA-Rolle mit einer Reihe gut getesteter ausgeliefert "Standardwerte" wird.

- b. Wählen Sie anschließend einen Fechten-Agent aus und konfigurieren Sie ihn:

- i. OPTION 1: Ermöglicht das Fechten mit 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>
"

```

- ii. OPTION 2: Ermöglicht das Fechten mit den vom Lenovo XCC (und anderen BMCs) bereitgestellten Redfish APIs:

```

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. Weitere Informationen zum Konfigurieren anderer Fencing-Agenten finden Sie im "[Red Hat-Dokumentation](#)".

10. Die BeeGFS HA-Rolle kann viele verschiedene Tuning-Parameter anwenden, um die Leistung weiter zu optimieren. Dazu gehören unter anderem die Optimierung der Kernel-Speicherauslastung und die E/A von Blockgeräten. Die Rolle wird mit einem angemessenen Satz von basierend auf Tests mit NetApp E-Series Block-Nodes ausgeliefert "[Standardwerte](#)". Diese werden standardmäßig jedoch nicht angewendet, es sei denn, Sie geben Folgendes an:

```
beegfs_ha_enable_performance_tuning: True
```

- a. Geben Sie bei Bedarf auch hier Änderungen an der Standard-Performance-Optimierung an. Weitere Informationen finden Sie in der vollständigen "[Parameter für die Performance-Optimierung](#)" Dokumentation.
11. Damit schwelende IP-Adressen (manchmal auch als logische Schnittstellen bekannt), die für BeeGFS-Dienste verwendet werden, zwischen Datei-Nodes ausfallen können, müssen alle Netzwerkschnittstellen konsistent benannt werden. Standardmäßig werden Netzwerkschnittstellennamen vom Kernel generiert, was nicht garantiert ist, dass konsistente Namen generiert werden, auch bei identischen Servermodellen mit Netzwerkadapters, die in denselben PCIe-Steckplätzen installiert sind. Dies ist auch nützlich, wenn Vorräte erstellt werden, bevor das Gerät bereitgestellt wird und generierte Schnittstellennamen bekannt sind. Um konsistente Gerätenamen auf der Grundlage eines Blockdiagramms des Servers oder sicherzustellen `lshw -class network -businfo` Ausgabe, geben Sie die gewünschte PCIe-Adresse-zu-logische Schnittstellenuordnung wie folgt an:

- a. Für InfiniBand (IBoI)-Netzwerkschnittstellen:

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

- b. Bei Ethernet-Netzwerkschnittstellen:

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



Um Konflikte zu vermeiden, wenn Schnittstellen umbenannt werden (um zu verhindern, dass sie umbenannt werden), sollten Sie keine möglichen Standardnamen wie eth0, ens9f0, ib0 oder ibs4f0 verwenden. Eine häufige Namenskonvention besteht darin, „e“ oder „i“ für Ethernet oder InfiniBand zu verwenden, gefolgt von der PCIe-Steckplatznummer und einem Buchstaben zur Angabe des Ports. Zum Beispiel wäre der zweite Port eines InfiniBand-Adapters, der in Steckplatz 3 installiert ist: i3b.



Wenn Sie ein verifiziertes Datei-Node-Modell verwenden, klicken Sie auf "[Hier](#)" Beispiel für Zuordnungen von PCIe-Adressen zu logischen Ports

12. Geben Sie optional die Konfiguration an, die für alle BeeGFS-Dienste im Cluster gelten soll. Die Standardkonfigurationswerte können gefunden werden "[Hier](#)", und die Konfiguration pro Service wird an anderer Stelle angegeben:

- a. BeeGFS Management-Service:

```
beegfs_ha_beeefs_mgmtd_conf_ha_group_options:  
<OPTION>: <VALUE>
```

- b. BeeGFS Metadata Services:

```
beegfs_ha_beeefs_meta_conf_ha_group_options:  
<OPTION>: <VALUE>
```

- c. BeeGFS Storage-Services:

```
beegfs_ha_beeefs_storage_conf_ha_group_options:  
<OPTION>: <VALUE>
```

13. Ab BeeGFS 7.2.7 und 7.3.1 "[Verbindungsauthentifizierung](#)" Muss konfiguriert oder explizit deaktiviert werden. Es gibt einige Konfigurationsmöglichkeiten, die mit der Ansible-basierten Implementierung konfiguriert werden können:

- a. Standardmäßig konfiguriert die Bereitstellung die Verbindungsauthentifizierung automatisch und erstellt ein `connauthfile` Die auf alle Datei-Nodes verteilt und mit den BeeGFS-Diensten verwendet werden. Diese Datei wird auch auf dem Ansible-Steuerungsknoten in abgelegt/gepflegt  
`<INVENTORY>/files/beegfs/<sysMgmtdHost>_connAuthFile` Wo Sie Daten für die Wiederverwendung mit Clients, die auf dieses Filesystem zugreifen müssen, aufbewahren (sicher).

- i. Zum Generieren eines neuen Schlüssels angeben -e  
`"beegfs_ha_conn_auth_force_new=True` Wenn Sie das Ansible-Playbook ausführen.  
Beachten Sie, dass dies bei einem ignoriert wird `beegfs_ha_conn_auth_secret` Definiert ist.
- ii. Weitere Optionen finden Sie in der vollständigen Liste der Standardwerte, die im enthalten

"BeeGFS HA-Rolle" sind.

- b. Ein benutzerdefiniertes Geheimnis kann verwendet werden, indem Sie Folgendes in definieren ha\_cluster.yml:

```
beegfs_ha_conn_auth_secret: <SECRET>
```

- c. Die Verbindungsauthentifizierung kann vollständig deaktiviert werden (NICHT empfohlen):

```
beegfs_ha_conn_auth_enabled: false
```

Klicken Sie Auf "[Hier](#)" Beispiel für eine komplette Bestandsdatei, die die allgemeine Konfiguration des Dateiknoten darstellt.

#### Verwendung von HDR (200 GB) InfiniBand mit NetApp EF600 Block-Nodes:

Um HDR (200 GB) InfiniBand mit der EF600 zu verwenden, muss der Subnetzmanager die Virtualisierung unterstützen. Wenn Datei- und Block-Knoten über einen Switch verbunden sind, muss dies im Subnetz Manager für die Gesamtstruktur aktiviert sein.

Wenn Block- und Datei-Nodes direkt über InfiniBand verbunden sind, opensm muss auf jedem Datei-Node für jede Schnittstelle, die direkt mit einem Block-Node verbunden ist, eine Instanz konfiguriert werden. Dies geschieht durch Angabe von `configure: true` wann "[Konfigurieren von File-Node-Storage-Schnittstellen](#)".

Derzeit unterstützt die Inbox-Version von `opensm`, die mit unterstützten Linux-Distributionen ausgeliefert wurde, keine Virtualisierung. Stattdessen ist es erforderlich, dass Sie die Version von über die NVIDIA OpenFabrics Enterprise Distribution (OFED) installieren und konfigurieren `opensm`. Obwohl die Implementierung mit Ansible weiterhin unterstützt wird, sind einige weitere Schritte erforderlich:

1. Laden Sie die Pakete für die Version von OpenSM, die im Abschnitt von der NVIDIA-Website aufgeführt sind, mithilfe von Curl oder Ihrem gewünschten Tool in das Verzeichnis herunter "[Technologieanforderungen erfüllt](#)" <INVENTORY>/packages/. Beispiel:

```
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/rhel9.4/x86_64/opensm-5.17.2.MLNX20240610.dc7c2998-
0.1.2310322.x86_64.rpm
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/rhel9.4/x86_64/opensm-libs-5.17.2.MLNX20240610.dc7c2998-
0.1.2310322.x86_64.rpm
```

2. Unter group\_vars/ha\_cluster.yml Definieren Sie die folgende Konfiguration:

```

### 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-5.17.2.MLNX20240610.dc7c2998-
0.1.2310322.x86_64.rpm": "/tmp/"
          "packages/opensm-libs-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"

```

## Festlegen Der Konfiguration Allgemeiner Blockknoten

Geben Sie unter Verwendung von Gruppenvariablen (Group\_vars) die allgemeine Konfiguration von Blockknoten an.

### Überblick

Die Konfiguration, die auf alle Block-Nodes Apfel soll, wird auf definiert group\_vars/eseries\_storage\_systems.yml. Dazu gehören in der Regel:

- Details dazu, wie der Ansible-Kontroll-Node mit als Block-Nodes verwendeten E-Series Storage-Systemen verbunden werden soll.

- Welche Firmware-, NVSRAM- und Laufwerk-Firmware-Versionen die Nodes ausführen sollten.
- Globale Konfiguration einschließlich Cache-Einstellungen, Host-Konfiguration und Einstellungen für die Bereitstellung von Volumes



Die in dieser Datei festgelegten Optionen können auch auf einzelnen Block-Nodes definiert werden, z. B. wenn gemischte Hardware-Modelle verwendet werden oder Sie unterschiedliche Passwörter für jeden Knoten haben. Die Konfiguration auf einzelnen Block-Knoten hat Vorrang vor der Konfiguration in dieser Datei.

## Schritte

Erstellen Sie die Datei `group_vars/eseries_storage_systems.yml` Und füllen Sie es wie folgt aus:

1. Ansible verwendet SSH nicht für die Verbindung mit Block-Nodes und verwendet stattdessen REST-APIs. Um dies zu erreichen, müssen wir Folgendes festlegen:

```
ansible_connection: local
```

2. Geben Sie den Benutzernamen und das Passwort an, um jeden Knoten zu verwalten. Der Benutzername kann optional ausgelassen werden (und wird standardmäßig auf admin gesetzt), andernfalls können Sie jedes Konto mit Administratorrechten angeben. Geben Sie außerdem an, ob SSL-Zertifikate überprüft oder ignoriert werden sollen:

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



Es wird nicht empfohlen, Kennwörter im Klartext zu verwenden. Verwenden Sie einen Ansible-Vault, oder stellen Sie die bereit `eseries_system_password` Wenn Sie Ansible mit --extra-Vars verwenden.

3. Geben Sie optional an, welche Controller-Firmware, NVSRAM und Laufwerk-Firmware auf den Nodes installiert werden soll. Diese müssen auf das heruntergeladen werden `packages/` Verzeichnis vor der Ausführung von Ansible. NVSRAM und E-Series Controller Firmware können heruntergeladen werden "Hier" Und Laufwerk-Firmware "Hier":

```

eseries_firmware_firmware: "packages/<FILENAME>.dlp" # Ex.
"packages/RCB_11.80GA_6000_64cc0ee3.dlp"
eseries_firmware_nvssram: "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".

```



Wenn diese Konfiguration angegeben wird, aktualisiert Ansible automatisch alle Firmware einschließlich des Neubootens von Controllern (falls erforderlich), ohne zusätzliche Eingabeaufforderungen. Dies wird für BeeGFS/Host-I/O voraussichtlich ohne Unterbrechung ausgeführt, kann jedoch zu einer vorübergehenden Abnahme der Performance führen.

4. Passen Sie die Standardeinstellungen für die globale Systemkonfiguration an. Die hier aufgeführten Optionen und Werte werden häufig für BeeGFS auf NetApp empfohlen, können jedoch bei Bedarf angepasst werden:

```

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. Konfigurieren Sie die Standardeinstellungen für die globale Volume-Bereitstellung. Die hier aufgeführten Optionen und Werte werden häufig für BeeGFS auf NetApp empfohlen, können jedoch bei Bedarf angepasst werden:

```

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. Passen Sie bei Bedarf die Reihenfolge an, in der Ansible Laufwerke für Storage Pools und Volume-Gruppen wählt, und berücksichtigen Sie die folgenden Best Practices:
  - a. Nennen Sie alle (möglicherweise kleineren) Laufwerke, die zuerst für Management- und/oder Metadaten-Volumes verwendet werden sollten, und Storage Volumes zuletzt.

- b. Stellen Sie sicher, dass Sie die Reihenfolge der Festplattenauswahl auf der Grundlage der Modelle für Festplatten-Shelfs/Festplattengehäuse ausgleichen, um die Laufwerksauswahl über verfügbare Laufwerkskanäle auszugleichen. Beispielsweise befinden sich Laufwerke 0-11 mit der EF600 und ohne Erweiterungen auf Laufwerkskanal 1 und Laufwerke 12-23 auf dem Laufwerkskanal. Daher ist eine Strategie zur Balance der Antriebsauswahl zu wählen disk shelf:drive 99:0, 99:23, 99:1, 99:22 usw. Wenn mehr als ein Gehäuse vorhanden ist, steht die erste Ziffer für die Laufwerk-Shelf-ID.

```
# 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"
```

Klicken Sie Auf "[Hier](#)" Beispiel für eine komplette Bestandsdatei, die die allgemeine Block-Node-Konfiguration darstellt.

## BeeGFS-Dienste definieren

### Definieren Sie den BeeGFS-Managementdienst

BeeGFS-Dienste werden mit Gruppenvariablen (Group\_vars) konfiguriert.

#### Überblick

In diesem Abschnitt wird die Definition des BeeGFS-Managementservice erläutert. In den HA-Clustern für ein bestimmtes Dateisystem sollte nur ein Service dieses Typs vorhanden sein. Die Konfiguration dieses Services umfasst die folgenden Punkte:

- Der Servicetyp (Management).
- Definieren von Konfigurationen, die nur für diesen BeeGFS-Dienst gelten sollen.
- Konfiguration einer oder mehrerer fließender IPs (logische Schnittstellen), an denen dieser Service erreicht werden kann.
- Geben Sie an, wo/wie ein Volume Daten für diesen Service speichern soll (das BeeGFS-Managementziel).

#### Schritte

Erstellen Sie eine neue Datei group\_vars/mgmt.yml, und verweisen Sie auf den "[Planen Sie das Dateisystem](#)" Abschnitt. Füllen Sie diese wie folgt aus:

1. Geben Sie diese Datei an, um die Konfiguration für einen BeeGFS-Managementdienst anzuzeigen:

```
beegfs_service: management
```

2. Definieren Sie alle Konfigurationen, die nur für diesen BeeGFS-Dienst gelten sollen. Dies ist in der Regel nicht für den Management-Service erforderlich, es sei denn, Sie müssen Quoten aktivieren, jedoch alle unterstützten Konfigurationsparameter von beegfs-mgmt.conf Kann enthalten sein. Beachten Sie, dass die folgenden Parameter automatisch/an anderer Stelle konfiguriert werden und hier nicht angegeben werden sollten: storeMgmtDirectory, connAuthFile, connDisableAuthentication,

connInterfacesFile, und connNetFilterFile.

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

3. Konfigurieren Sie eine oder mehrere unverankerte IPs, die andere Dienste und Clients verwenden, um eine Verbindung zu diesem Dienst herzustellen (dadurch wird das BeeGFS automatisch festgelegt connInterfacesFile Option):

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

4. Geben Sie optional ein oder mehrere zulässige IP-Subnetze an, die für die ausgehende Kommunikation verwendet werden können (dadurch wird automatisch das BeeGFS eingestellt connNetFilterFile Option):

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

5. Geben Sie das BeeGFS-Managementziel an, auf dem dieser Service Daten gemäß den folgenden Richtlinien speichert:

- Für mehrere BeeGFS Services/Ziele kann derselbe Speicherpool oder Volume-Gruppenname verwendet werden. Stellen Sie einfach sicher, dass er dasselbe verwendet name, raid\_level, criteria\_\*, und common\_\* Konfiguration für jede einzelne (die für jeden Service aufgeführten Volumes sollten unterschiedlich sein).
- Volume-Größen sollten als Prozentsatz der Storage-Pool/Volume-Gruppe angegeben werden. Die Summe sollte bei allen Services/Volumes, die über einen bestimmten Storage-Pool/Volume-Gruppe verfügen, nicht mehr als 100 übersteigen. Hinweis: Bei der Verwendung von SSDs wird empfohlen, freien Speicherplatz in der Volume-Gruppe zu belassen, um die SSD-Performance und den SSD-Verschleiß ["Hier"](#) zu maximieren (klicken Sie für weitere Details).
- Klicken Sie Auf ["Hier"](#) Eine vollständige Liste der für das verfügbaren Konfigurationsoptionen finden Sie unter eseries\_storage\_pool\_configuration. Notieren Sie einige Optionen wie z. B. state, host, host\_type, workload\_name, und workload\_metadata Und Volume-Namen werden automatisch generiert und sollten hier nicht angegeben werden.

```

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

```

Klicken Sie Auf "[Hier](#)" Beispiel für eine komplette Bestandsdatei, die einen BeeGFS-Managementdienst darstellt.

## Definieren Sie den BeeGFS-Metadatendienst

BeeGFS-Dienste werden mit Gruppenvariablen (Group\_vars) konfiguriert.

### Überblick

In diesem Abschnitt wird die Definition des BeeGFS-Metadatendienstes erläutert. In den HA-Clustern für ein bestimmtes Dateisystem sollte mindestens ein Service dieses Typs vorhanden sein. Die Konfiguration dieses Services umfasst die folgenden Punkte:

- Der Servicetyp (Metadaten).
- Definieren von Konfigurationen, die nur für diesen BeeGFS-Dienst gelten sollen.
- Konfiguration einer oder mehrerer fließender IPs (logische Schnittstellen), an denen dieser Service erreicht werden kann.
- Festlegen, wo/wie ein Volume Daten für diesen Service speichern soll (das BeeGFS-Metadatenziel).

### Schritte

["Planen Sie das Dateisystem"](#) Erstellen Sie group\_vars/meta\_<ID>.yml für jeden Metadatendienst im Cluster eine Datei unter, und füllen Sie sie wie folgt aus, um auf den Abschnitt zu verweisen:

1. Geben Sie an, dass diese Datei die Konfiguration für einen BeeGFS-Metadatendienst darstellt:

```
beegfs_service: metadata
```

2. Definieren Sie alle Konfigurationen, die nur für diesen BeeGFS-Dienst gelten sollen. Mindestens müssen Sie den gewünschten TCP- und UDP-Port angeben, jedoch alle unterstützten Konfigurationsparameter von beegfs-meta.conf Kann ebenfalls enthalten sein. Beachten Sie, dass die folgenden Parameter automatisch/an anderer Stelle konfiguriert werden und hier nicht angegeben werden sollten:  
sysMgmtHost, storeMetaDirectory, connAuthFile, connDisableAuthentication,

connInterfacesFile, und 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. Konfigurieren Sie eine oder mehrere unverankerte IPs, die andere Dienste und Clients verwenden, um eine Verbindung zu diesem Dienst herzustellen (dadurch wird das BeeGFS automatisch festgelegt connInterfacesFile Option):

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

4. Geben Sie optional ein oder mehrere zulässige IP-Subnetze an, die für die ausgehende Kommunikation verwendet werden können (dadurch wird automatisch das BeeGFS eingestellt connNetFilterFile Option):

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

5. Geben Sie das BeeGFS-Metadatenziel an, bei dem dieser Dienst Daten gemäß den folgenden Richtlinien speichert (dies konfiguriert auch automatisch den storeMetaDirectory Option):
- Für mehrere BeeGFS Services/Ziele kann derselbe Speicherpool oder Volume-Gruppenname verwendet werden. Stellen Sie einfach sicher, dass er dasselbe verwendet name, raid\_level, criteria\_\*, und common\_\* Konfiguration für jede einzelne (die für jeden Service aufgeführten Volumes sollten unterschiedlich sein).
  - Volume-Größen sollten als Prozentsatz der Storage-Pool/Volume-Gruppe angegeben werden. Die Summe sollte bei allen Services/Volumes, die über einen bestimmten Storage-Pool/Volume-Gruppe verfügen, nicht mehr als 100 übersteigen. Hinweis: Bei der Verwendung von SSDs wird empfohlen, freien Speicherplatz in der Volume-Gruppe zu belassen, um die SSD-Performance und den SSD-Verschleiß "Hier" zu maximieren (klicken Sie für weitere Details).
  - Klicken Sie Auf "Hier" Eine vollständige Liste der für das verfügbaren Konfigurationsoptionen finden Sie unter eseries\_storage\_pool\_configuration. Notieren Sie einige Optionen wie z. B. state, host, host\_type, workload\_name, und workload\_metadata Und Volume-Namen werden automatisch generiert und sollten hier nicht angegeben werden.

```

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

```

Klicken Sie Auf "[Hier](#)" Beispiel für eine komplette Bestandsdatei, die einen BeeGFS-Metadatendienst darstellt.

## Definieren Sie den BeeGFS-Speicherdienst

BeeGFS-Dienste werden mit Gruppenvariablen (Group\_vars) konfiguriert.

### Überblick

In diesem Abschnitt wird die Definition des BeeGFS-Speicherdienstes erläutert. In den HA-Clustern für ein bestimmtes Dateisystem sollte mindestens ein Service dieses Typs vorhanden sein. Die Konfiguration dieses Services umfasst die folgenden Punkte:

- Den Servicetyp (Storage).
- Definieren von Konfigurationen, die nur für diesen BeeGFS-Dienst gelten sollen.
- Konfiguration einer oder mehrerer fließender IPs (logische Schnittstellen), an denen dieser Service erreicht werden kann.
- Geben Sie an, wo/wie Volumen(en) Daten für diesen Dienst speichern sollen (die BeeGFS-Speicherziele).

### Schritte

["Planen Sie das Dateisystem"](#) Erstellen Sie group\_vars/stor\_<ID>.yml für jeden Storage-Service im Cluster eine Datei unter, und füllen Sie sie wie folgt aus, um auf den Abschnitt Bezug zu nehmen:

1. Geben Sie diese Datei für die Konfiguration eines BeeGFS-Speicherdienstes an:

```
beegfs_service: storage
```

2. Definieren Sie alle Konfigurationen, die nur für diesen BeeGFS-Dienst gelten sollen. Mindestens müssen Sie den gewünschten TCP- und UDP-Port angeben, jedoch alle unterstützten Konfigurationsparameter von beegfs-storage.conf Kann ebenfalls enthalten sein. Beachten Sie, dass die folgenden Parameter automatisch/an anderer Stelle konfiguriert werden und hier nicht angegeben werden sollten: sysMgmtHost, storeStorageDirectory, connAuthFile, connDisableAuthentication, connInterfacesFile, und connNetFilterFile.

```

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

```

3. Konfigurieren Sie eine oder mehrere unverankerte IPs, die andere Dienste und Clients verwenden, um eine Verbindung zu diesem Dienst herzustellen (dadurch wird das BeeGFS automatisch festgelegt `connInterfacesFile` Option):

```

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

```

4. Geben Sie optional ein oder mehrere zulässige IP-Subnetze an, die für die ausgehende Kommunikation verwendet werden können (dadurch wird automatisch das BeeGFS eingestellt `connNetFilterFile` Option):

```

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

```

5. Geben Sie die BeeGFS-Speicherziele an, in denen dieser Service Daten gemäß den folgenden Richtlinien speichert (dies konfiguriert auch automatisch den `storeStorageDirectory` Option):

- Für mehrere BeeGFS Services/Ziele kann dieselbe Speicherpool oder Volume-Gruppenname verwendet werden. Stellen Sie einfach sicher, dass er dasselbe verwendet `name`, `raid_level`, `criteria_*`, und `common_*` Konfiguration für jede einzelne (die für jeden Service aufgeführten Volumes sollten unterschiedlich sein).
- Volume-Größen sollten als Prozentsatz der Storage-Pool/Volume-Gruppe angegeben werden. Die Summe sollte bei allen Services/Volumes, die über einen bestimmten Storage-Pool/Volume-Gruppe verfügen, nicht mehr als 100 übersteigen. Hinweis: Bei der Verwendung von SSDs wird empfohlen, freien Speicherplatz in der Volume-Gruppe zu belassen, um die SSD-Performance und den SSD-Verschleiß "[Hier](#)" zu maximieren (klicken Sie für weitere Details).
- Klicken Sie Auf "[Hier](#)" Eine vollständige Liste der für das verfügbaren Konfigurationsoptionen finden Sie unter `eseries_storage_pool_configuration`. Notieren Sie einige Optionen wie z. B. `state`, `host`, `host_type`, `workload_name`, und `workload_metadata` Und Volume-Namen werden automatisch generiert und sollten hier nicht angegeben werden.

```

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

```

Klicken Sie Auf "[Hier](#)" Beispiel für eine komplette Bestandsdatei, die einen BeeGFS-Speicherdiens darstellt.

## Zuordnen von BeeGFS-Services zu Datei-Nodes

Geben Sie an, welche Datei-Nodes jeden BeeGFS-Dienst mit dem ausführen können  
inventory.yml Datei:

### Überblick

In diesem Abschnitt wird die Erstellung des erläutert inventory.yml Datei: Dazu gehören alle Block-Nodes und die Angabe, welche Datei-Nodes jeden BeeGFS-Service ausführen können.

### Schritte

Erstellen Sie die Datei inventory.yml Und füllen Sie es wie folgt aus:

1. Erstellen Sie oben in der Datei die Ansible-Standardinventarstruktur:

```

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

```

2. Erstellen Sie eine Gruppe mit allen Block-Nodes, die an diesem HA-Cluster teilnehmen:

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

3. Erstellen Sie eine Gruppe, die alle BeeGFS-Dienste im Cluster und die Datei-Nodes enthält, auf denen sie ausgeführt werden:

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

4. Definieren Sie für jeden BeeGFS-Service im Cluster die bevorzugten und sekundären Dateiknoten, die diesen Service ausführen sollen:

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

Klicken Sie Auf "[Hier](#)" Beispiel für eine komplette Bestandsdatei.

## **Copyright-Informationen**

Copyright © 2026 NetApp. Alle Rechte vorbehalten. Gedruckt in den USA. Dieses urheberrechtlich geschützte Dokument darf ohne die vorherige schriftliche Genehmigung des Urheberrechtsinhabers in keiner Form und durch keine Mittel – weder grafische noch elektronische oder mechanische, einschließlich Fotokopieren, Aufnehmen oder Speichern in einem elektronischen Abrufsystem – auch nicht in Teilen, vervielfältigt werden.

Software, die von urheberrechtlich geschütztem NetApp Material abgeleitet wird, unterliegt der folgenden Lizenz und dem folgenden Haftungsausschluss:

DIE VORLIEGENDE SOFTWARE WIRD IN DER VORLIEGENDEN FORM VON NETAPP ZUR VERFÜGUNG GESTELLT, D. H. OHNE JEGLICHE EXPLIZITE ODER IMPLIZITE GEWÄHRLEISTUNG, EINSCHLIESSLICH, JEDOCH NICHT BESCHRÄNKKT AUF DIE STILLSCHWEIGENDE GEWÄHRLEISTUNG DER MARKTGÄNGIGKEIT UND EIGNUNG FÜR EINEN BESTIMMTEN ZWECK, DIE HIERMIT AUSGESCHLOSSEN WERDEN. NETAPP ÜBERNIMMT KEINERLEI HAFTUNG FÜR DIREKTE, INDIREKTE, ZUFÄLLIGE, BESONDERE, BEISPIELHAFFE SCHÄDEN ODER FOLGESCHÄDEN (EINSCHLIESSLICH, JEDOCH NICHT BESCHRÄNKKT AUF DIE BESCHAFFUNG VON ERSATZWAREN ODER -DIENSTLEISTUNGEN, NUTZUNGS-, DATEN- ODER GEWINNVERLUSTE ODER UNTERBRECHUNG DES GESCHÄFTSBETRIEBS), UNABHÄNGIG DAVON, WIE SIE VERURSACHT WURDEN UND AUF WELCHER HAFTUNGSTHEORIE SIE BERUHEN, OB AUS VERTRAGLICH FESTGELEGTER HAFTUNG, VERSCHULDENSUNABHÄNGIGER HAFTUNG ODER DELIKTSHAFTUNG (EINSCHLIESSLICH FAHRLÄSSIGKEIT ODER AUF ANDEREM WEGE), DIE IN IRGENDERWEINER WEISE AUS DER NUTZUNG DIESER SOFTWARE RESULTIEREN, SELBST WENN AUF DIE MÖGLICHKEIT DERARTIGER SCHÄDEN HINGEWIESEN WURDE.

NetApp behält sich das Recht vor, die hierin beschriebenen Produkte jederzeit und ohne Vorankündigung zu ändern. NetApp übernimmt keine Verantwortung oder Haftung, die sich aus der Verwendung der hier beschriebenen Produkte ergibt, es sei denn, NetApp hat dem ausdrücklich in schriftlicher Form zugestimmt. Die Verwendung oder der Erwerb dieses Produkts stellt keine Lizenzierung im Rahmen eines Patentrechts, Markenrechts oder eines anderen Rechts an geistigem Eigentum von NetApp dar.

Das in diesem Dokument beschriebene Produkt kann durch ein oder mehrere US-amerikanische Patente, ausländische Patente oder anhängige Patentanmeldungen geschützt sein.

**ERLÄUTERUNG ZU „RESTRICTED RIGHTS“:** Nutzung, Vervielfältigung oder Offenlegung durch die US-Regierung unterliegt den Einschränkungen gemäß Unterabschnitt (b)(3) der Klausel „Rights in Technical Data – Noncommercial Items“ in DFARS 252.227-7013 (Februar 2014) und FAR 52.227-19 (Dezember 2007).

Die hierin enthaltenen Daten beziehen sich auf ein kommerzielles Produkt und/oder einen kommerziellen Service (wie in FAR 2.101 definiert) und sind Eigentum von NetApp, Inc. Alle technischen Daten und die Computersoftware von NetApp, die unter diesem Vertrag bereitgestellt werden, sind gewerblicher Natur und wurden ausschließlich unter Verwendung privater Mittel entwickelt. Die US-Regierung besitzt eine nicht ausschließliche, nicht übertragbare, nicht unterlizenzierbare, weltweite, limitierte unwiderrufliche Lizenz zur Nutzung der Daten nur in Verbindung mit und zur Unterstützung des Vertrags der US-Regierung, unter dem die Daten bereitgestellt wurden. Sofern in den vorliegenden Bedingungen nicht anders angegeben, dürfen die Daten ohne vorherige schriftliche Genehmigung von NetApp, Inc. nicht verwendet, offengelegt, vervielfältigt, geändert, aufgeführt oder angezeigt werden. Die Lizenzrechte der US-Regierung für das US-Verteidigungsministerium sind auf die in DFARS-Klausel 252.227-7015(b) (Februar 2014) genannten Rechte beschränkt.

## **Markeninformationen**

NETAPP, das NETAPP Logo und die unter <http://www.netapp.com/TM> aufgeführten Marken sind Marken von NetApp, Inc. Andere Firmen und Produktnamen können Marken der jeweiligen Eigentümer sein.