Examen du design de la solution

BeeGFS on NetApp with E-Series Storage

NetApp
August 23, 2024

Sommaire

Examen du design de la solution

Présentation du design

Pour prendre en charge la solution BeeGFS sur NetApp, qui associe le système de fichiers parallèle BeeGFS et les systèmes de stockage NetApp EF600, vous devez utiliser un équipement, un câblage et des configurations spécifiques.

En savoir plus :

- "Configuration matérielle"
- "Configuration logicielle"
- "Vérification de la conception"
- "Instructions de dimensionnement"
- "Réglage des performances"

Architectures dérivées avec des variations de conception et de performances :

- "Élément de base haute capacité"

Configuration matérielle

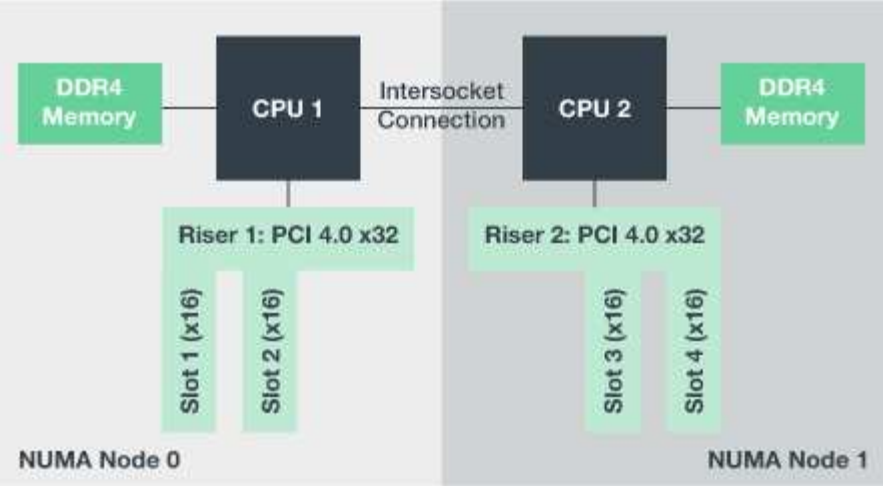
La configuration matérielle de BeeGFS sur NetApp inclut des nœuds de fichiers et le câblage réseau.

Configuration de nœud de fichiers

Les nœuds de fichiers ont deux sockets de CPU configurés en zones NUMA distinctes, qui incluent un accès local à un nombre égal de slots PCIe et de mémoire.

Les adaptateurs InfiniBand doivent être placés dans les connecteurs ou les cartes de montage PCI appropriés, de sorte que la charge de travail soit équilibrée sur les voies PCIe et les canaux de mémoire disponibles. Pour équilibrer la charge de travail, vous pouvez isoler intégralement le travail des services BeeGFS vers un nœud NUMA particulier. L'objectif est d'atteindre les mêmes performances pour chaque nœud de fichiers que s'il s'agissait de deux serveurs à socket unique indépendants.

La figure suivante montre la configuration NUMA du nœud de fichiers.



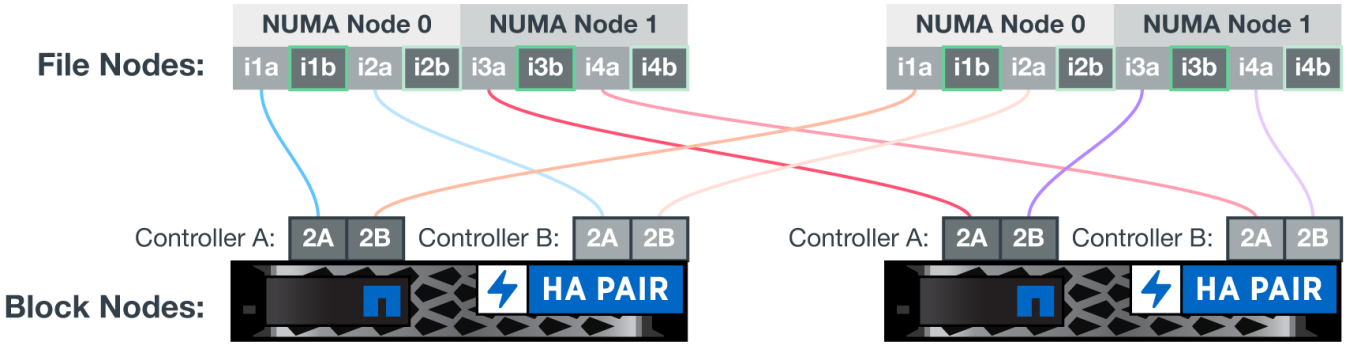
Les processus BeeGFS sont épinglés à une zone NUMA particulière pour s'assurer que les interfaces utilisées se trouvent dans la même zone. Cette configuration évite d'avoir besoin d'un accès à distance via la connexion inter-socket. La connexion inter-sockets est parfois appelée liaison QPI ou GMI2 ; même dans les architectures de processeurs modernes, ils peuvent être un goulot d'étranglement lors de l'utilisation de réseaux haut débit comme HDR InfiniBand.

Configuration des câbles réseau

Dans un élément de base, chaque nœud de fichier est connecté à deux nœuds de bloc grâce à quatre connexions InfiniBand redondantes. En outre, chaque nœud de fichiers dispose de quatre connexions redondantes au réseau de stockage InfiniBand.

Dans la figure suivante, notez que :

- Tous les ports de nœuds de fichiers indiqués en vert sont utilisés pour la connexion au maillage Storage Fabric ; tous les autres ports de nœuds de fichiers sont les connexions directes aux nœuds de blocs.
- Deux ports InfiniBand d'une zone NUMA spécifique se connectent aux contrôleurs A et B du même nœud de bloc.
- Les ports du nœud NUMA 0 se connectent toujours au premier nœud de bloc.
- Les ports du nœud NUMA 1 se connectent au second nœud de bloc.



Lors de l'utilisation de câbles de séparation pour connecter le commutateur de stockage aux nœuds de fichiers, un câble doit se brancher et se connecter aux ports indiqués en vert clair. Un autre câble doit se brancher et se connecter aux ports indiqués en vert foncé. En outre, pour les réseaux de stockage avec commutateurs redondants, les ports indiqués en vert clair doivent se connecter à un commutateur, tandis que les ports en vert foncé doivent se connecter à un autre commutateur.

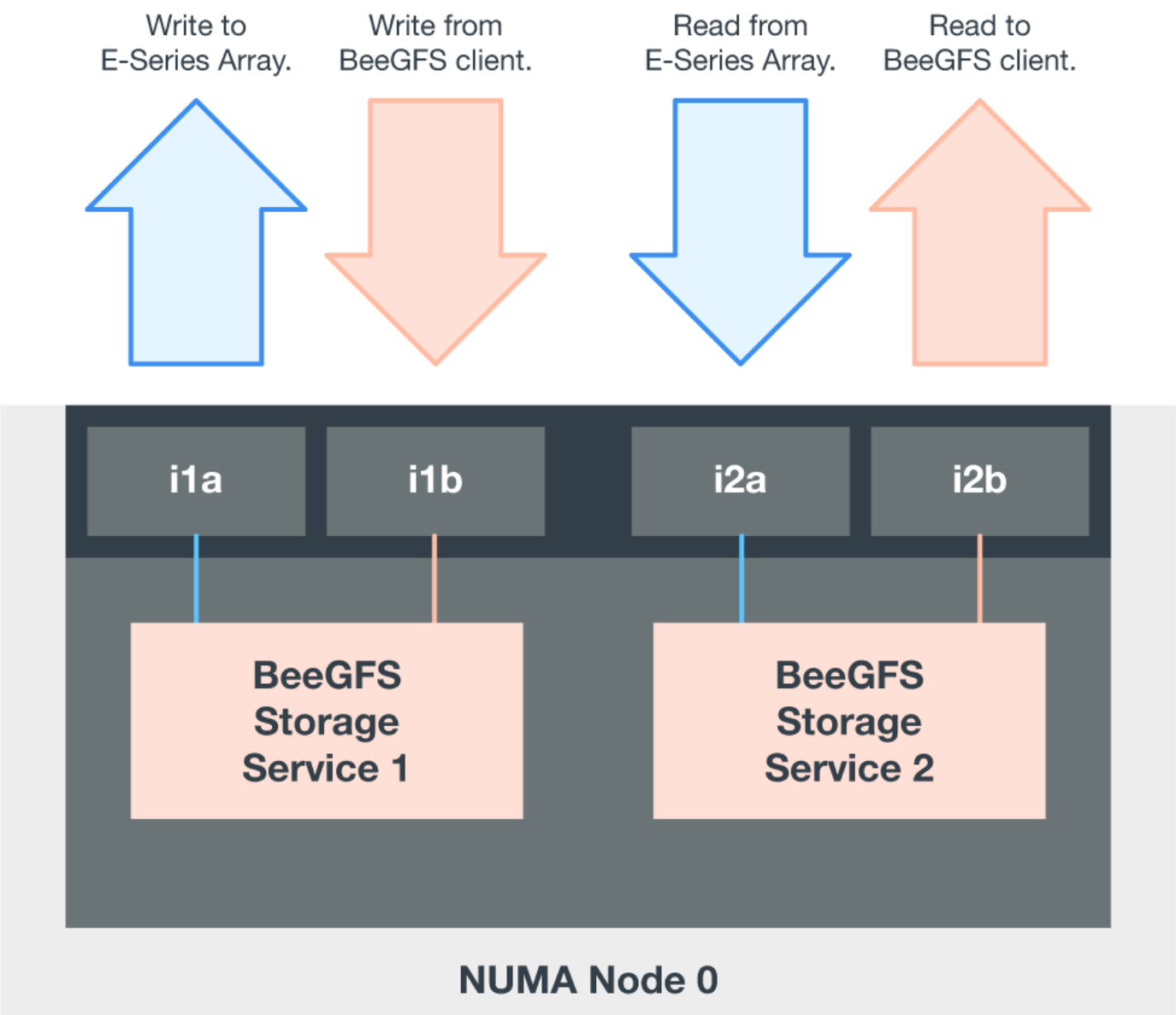
La configuration de câblage illustrée dans la figure permet à chaque service BeeGFS de :

- Exécuter dans la même zone NUMA, quel que soit le nœud de fichier qui exécute le service BeeGFS.
- Disposer de chemins optimaux secondaires au réseau de stockage frontal et aux nœuds de blocs internes, quel que soit l'endroit où une défaillance se produit
- Réduisez l'impact sur la performance si un nœud de fichiers ou un contrôleur d'un nœud de blocs nécessite une maintenance.

Câblage pour exploiter la bande passante

Pour exploiter la bande passante bidirectionnelle PCIe complète, vérifiez que un port de chaque adaptateur InfiniBand se connecte à la structure de stockage, et que l'autre port est connecté à un nœud de bloc.

La figure suivante montre la conception de câblage utilisée pour exploiter la bande passante bidirectionnelle PCIe complète.



Pour chaque service BeeGFS, utilisez la même carte pour connecter le port préféré utilisé pour le trafic client avec le chemin vers le contrôleur de nœuds de bloc qui est le principal propriétaire de ces volumes de services. Pour plus d'informations, voir "Configuration logicielle".

Configuration logicielle

La configuration logicielle de BeeGFS sur NetApp inclut des composants réseau BeeGFS, des nœuds de bloc EF600, des nœuds de fichiers BeeGFS, des groupes de ressources et des services BeeGFS.

Configuration réseau BeeGFS

La configuration du réseau BeeGFS comprend les composants suivants.

- **IP flottantes** les adresses IP flottantes sont un type d'adresse IP virtuelle qui peut être routée dynamiquement vers n'importe quel serveur du même réseau. Plusieurs serveurs peuvent posséder la même adresse IP flottante, mais elle ne peut être active que sur un seul serveur à la fois.

Chaque service de serveur BeeGFS possède sa propre adresse IP qui peut se déplacer entre les nœuds de fichiers en fonction de l'emplacement d'exécution du service de serveur BeeGFS. Cette configuration IP flottante permet à chaque service de basculer indépendamment vers l'autre nœud de fichiers. Le client a simplement besoin de connaître l'adresse IP d'un service BeeGFS particulier; il n'est pas nécessaire de savoir quel nœud de fichier exécute actuellement ce service.

- **Configuration multi-homing du serveur BeeGFS** pour augmenter la densité de la solution, chaque nœud de fichiers a plusieurs interfaces de stockage avec des adresses IP configurées dans le même sous-réseau IP.

  Des configurations supplémentaires sont nécessaires pour s'assurer que cette configuration fonctionne comme prévu avec la pile réseau Linux, car par défaut, les requêtes à une interface peuvent être traitées sur une autre interface si leurs adresses IP se trouvent dans le même sous-réseau. Outre d'autres inconvénients, ce comportement par défaut rend impossible l'établissement ou la maintenance des connexions RDMA.

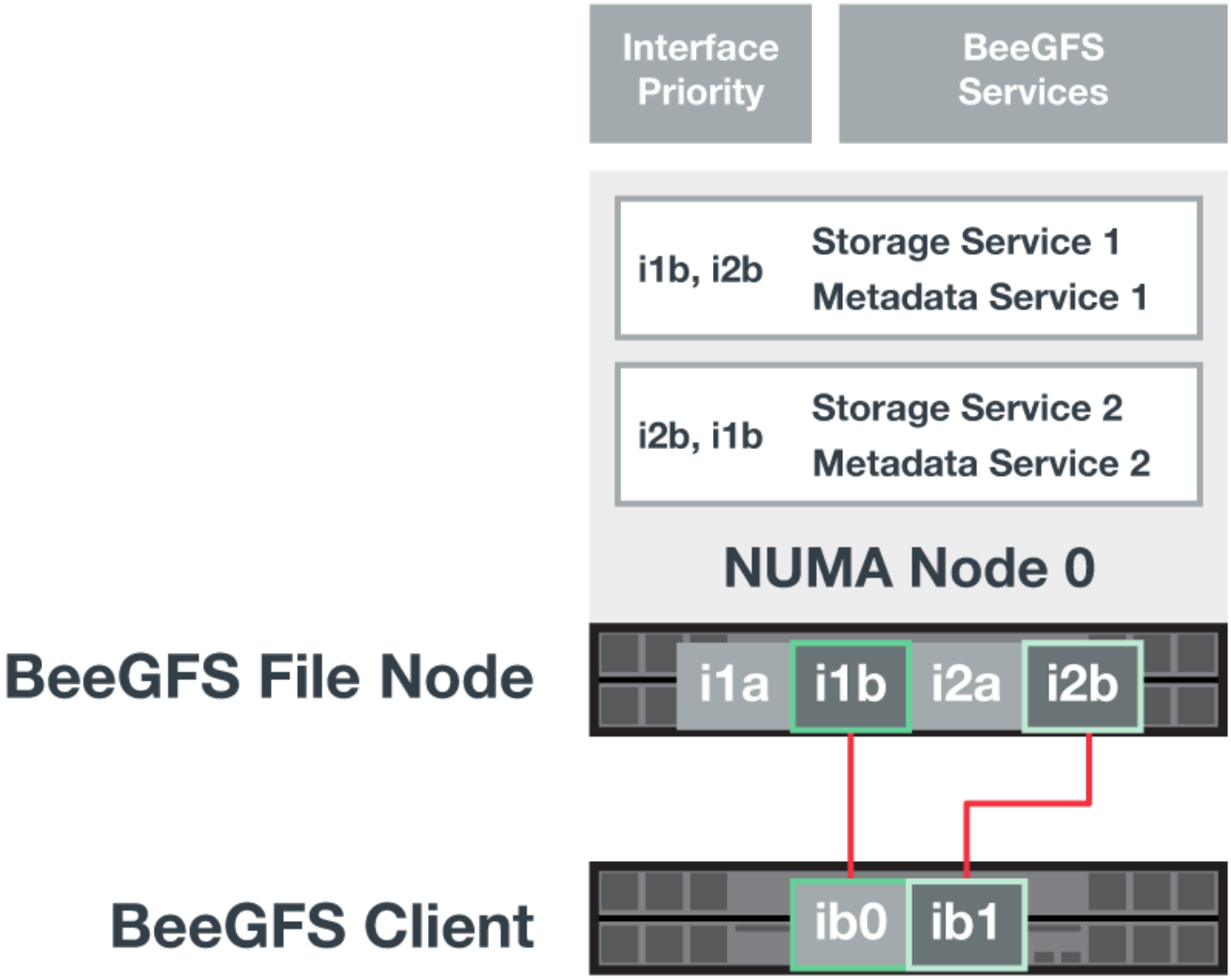
  Le déploiement Ansible gère le serrage du comportement de la RP (reverse path) et du protocole ARP (Address Resolution Protocol), ainsi que la vérification du démarrage et de l'arrêt d'adresses IP flottantes ; les routes et règles IP correspondantes sont créées dynamiquement pour permettre à la configuration réseau multihomée de fonctionner correctement.

- **La configuration multirail du client BeeGFS** *Multi-rail* fait référence à la capacité d'une application à utiliser plusieurs connexions réseau indépendantes, ou « rails », pour améliorer les performances.

  BeeGFS implémente la prise en charge multirail afin de permettre l'utilisation de plusieurs interfaces IB dans un seul sous-réseau IPoIB. Cette fonctionnalité permet notamment l'équilibrage dynamique de la charge entre les cartes réseau RDMA, optimisant ainsi l'utilisation des ressources du réseau. Il s'intègre également au système de stockage NVIDIA GPUDirect (GDS), qui offre une bande passante système accrue et réduit la latence et l'utilisation sur le processeur du client.

  Cette documentation fournit des instructions pour les configurations de sous-réseau IPoIB uniques. Les configurations de sous-réseau Dual IPoIB sont prises en charge, mais ne fournissent pas les mêmes avantages que les configurations à sous-réseau unique.

  La figure suivante montre l'équilibrage du trafic sur plusieurs interfaces client BeeGFS.



Comme chaque fichier de BeeGFS est généralement réparti sur plusieurs services de stockage, la configuration multi-rail permet au client d'atteindre un débit supérieur à celui d'un seul port InfiniBand. Par exemple, l'exemple de code suivant montre une configuration commune de répartition des fichiers qui permet au client d'équilibrer le trafic entre les deux interfaces :

+

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

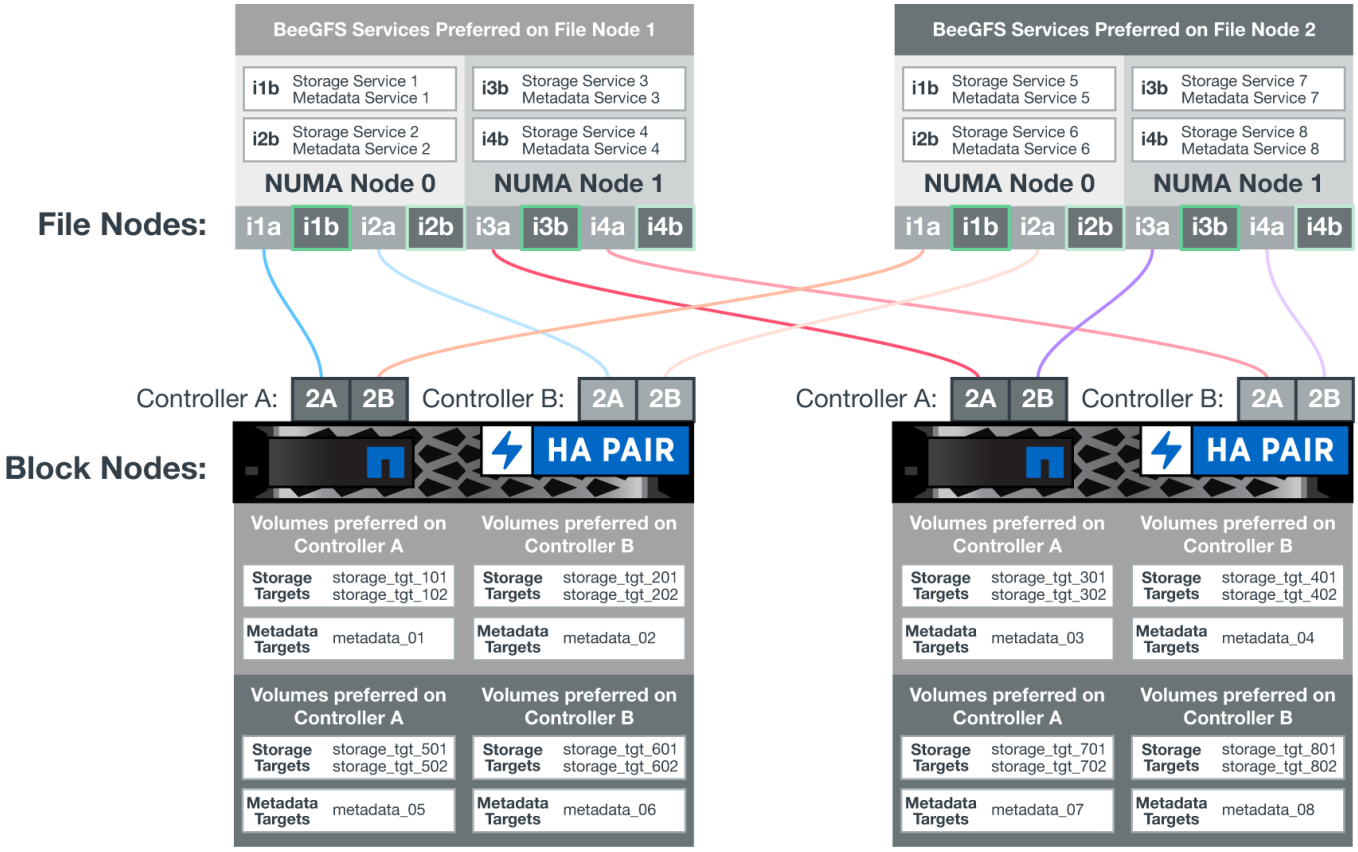
Configuration de nœud en mode bloc EF600

Les nœuds de blocs comprennent deux contrôleurs RAID actifs/actifs avec accès partagé au même ensemble de lecteurs. En général, chaque contrôleur possède la moitié des volumes configurés sur le système, mais il peut reprendre l'autre contrôleur si nécessaire.

Le logiciel de chemins d'accès multiples des nœuds de fichiers détermine le chemin actif et optimisé vers chaque volume et le déplace automatiquement vers l'autre chemin d'accès en cas de défaillance du câble, de l'adaptateur ou du contrôleur.

Le schéma suivant illustre la disposition du contrôleur dans les nœuds de bloc EF600.

Pour faciliter la solution haute disponibilité du disque partagé, les volumes sont mappés sur les deux nœuds de fichiers de manière à ce qu'ils puissent prendre en charge les uns les autres selon les besoins. Le diagramme suivant montre un exemple de configuration du service BeeGFS et de la propriété du volume préféré pour des performances maximales. L'interface à gauche de chaque service BeeGFS indique l'interface préférée que les clients et les autres services utilisent pour le contacter.



Dans l'exemple précédent, les clients et les services serveur préfèrent communiquer avec le service de stockage 1 via l'interface i1b. Le service de stockage 1 utilise l'interface i1a comme chemin préféré pour communiquer avec ses volumes (Storage_tgt_101, 102) sur le contrôleur A du premier nœud de bloc. Cette configuration utilise la bande passante PCIe bidirectionnelle complète disponible pour l'adaptateur InfiniBand et offre de meilleures performances avec un adaptateur HDR InfiniBand à deux ports que le tout avec PCIe 4.0.

Configuration de nœud de fichier BeeGFS

Les nœuds de fichiers BeeGFS sont configurés dans un cluster haute disponibilité (HA) pour faciliter le basculement des services BeeGFS entre plusieurs nœuds de fichiers.

La conception du cluster HA repose sur deux projets Linux HA largement utilisés : Corosync pour l'appartenance à un cluster et Pacemaker pour la gestion des ressources de cluster. Pour plus d'informations, voir "Formation Red Hat pour les modules complémentaires haute disponibilité".

NetApp a rédigé et étendu plusieurs agents de ressources OCF (Open Cluster Framework) pour permettre au cluster de démarrer et de surveiller intelligemment les ressources BeeGFS.

Clusters HA BeeGFS

De façon générale, lorsque vous démarrez un service BeeGFS (avec ou sans HA), quelques ressources doivent être en place :

- Adresses IP où le service est accessible, généralement configurées par Network Manager.
- Les systèmes de fichiers sous-jacents sont utilisés comme cibles de BeeGFS pour stocker des données.

Celles-ci sont généralement définies dans `/etc/fstab` Et monté par systemd.

- Un service systemd responsable du démarrage des processus BeeGFS lorsque les autres ressources sont prêtes.

  Sans logiciel supplémentaire, ces ressources ne démarrent que sur un seul nœud de fichiers. Par conséquent, si le nœud de fichier passe hors ligne, une partie du système de fichiers BeeGFS est inaccessible.

Comme plusieurs nœuds peuvent démarrer chaque service BeeGFS, Pacemaker doit s'assurer que chaque service et chaque ressource dépendante sont exécutés sur un seul nœud à la fois. Par exemple, si deux nœuds tentent de démarrer le même service BeeGFS, il y a un risque de corruption des données s'ils essaient tous les deux d'écrire sur les mêmes fichiers sur la cible sous-jacente. Pour éviter ce scénario, Pacemaker utilise Corosync pour maintenir en toute fiabilité l'état du cluster global en mode synchrone sur tous les nœuds et établir le quorum.

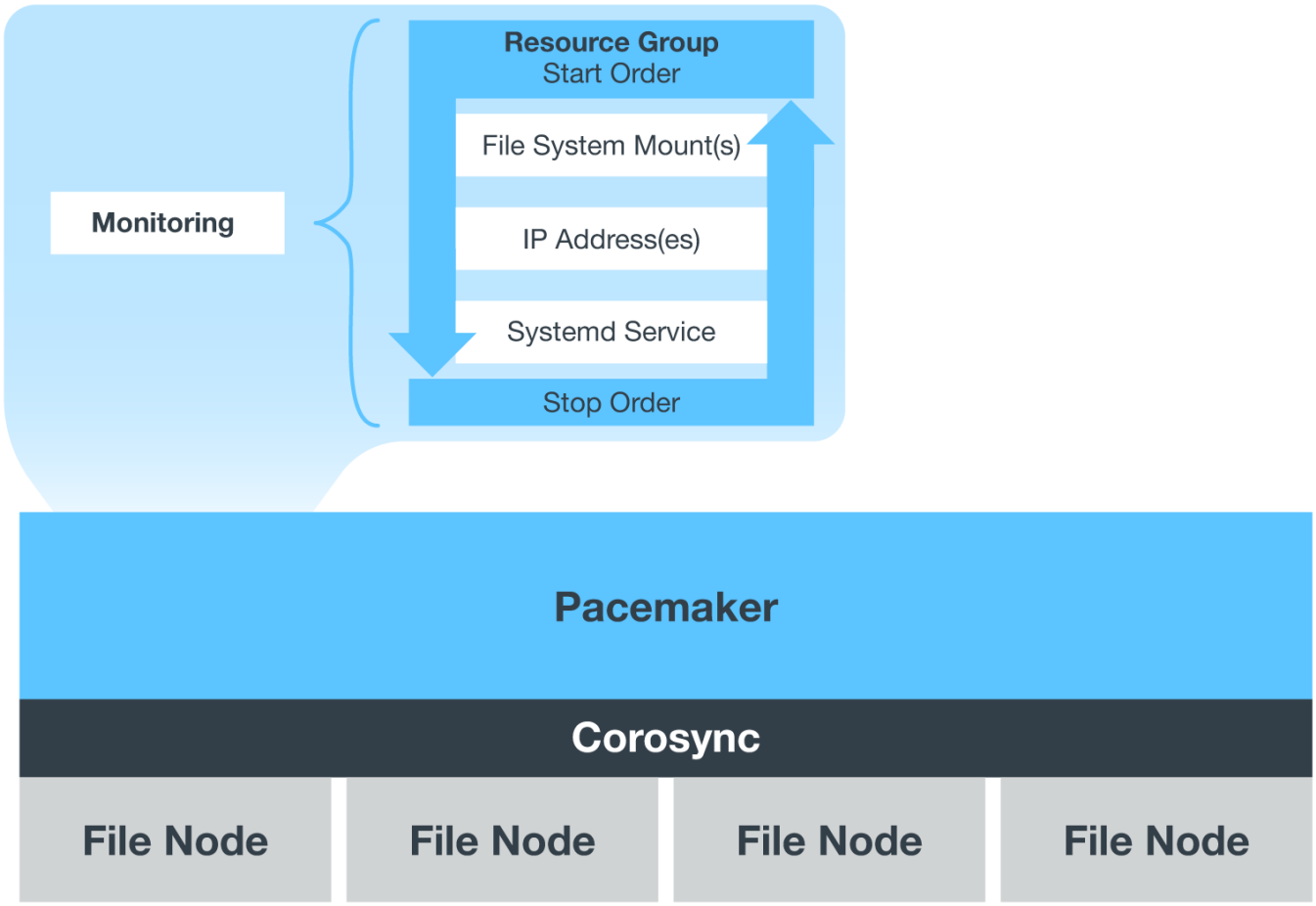
En cas de défaillance dans le cluster, Pacemaker réagit et redémarre les ressources BeeGFS sur un autre nœud. Dans certains cas, il se peut que Pacemaker ne puisse pas communiquer avec le nœud défectueux d'origine pour confirmer que les ressources sont arrêtées. Pour vérifier que le nœud est arrêté avant de redémarrer les ressources BeeGFS ailleurs, Pacemaker déligne le nœud défectueux, idéalement en retirant l'alimentation.

De nombreux agents d'escrime open source sont disponibles pour permettre à Pacemaker de verrouiller un nœud avec une unité de distribution d'alimentation (PDU) ou à l'aide du contrôleur BMC (Baseboard Management Controller) de serveur avec des API telles que Redfish.

Lorsque BeeGFS est exécuté dans un cluster HA, tous les services BeeGFS et les ressources sous-jacentes sont gérés par Pacemaker dans des groupes de ressources. Chaque service BeeGFS et les ressources dont il dépend sont configurés dans un groupe de ressources qui assure le démarrage et l'arrêt des ressources dans le bon ordre et qui sont situés sur le même nœud.

Pour chaque groupe de ressources BeeGFS, Pacemaker exécute une ressource de surveillance BeeGFS personnalisée qui est chargée de détecter les conditions de défaillance et de déclencher intelligemment les basculements lorsqu'un service BeeGFS n'est plus accessible sur un nœud particulier.

La figure suivante montre les services et les dépendances de BeeGFS contrôlés par Pacemaker.



Pour que plusieurs services BeeGFS du même type soient démarrés sur le même nœud, Pacemaker est configuré pour démarrer les services BeeGFS à l'aide de la méthode de configuration Multi-mode. Pour plus d'informations, reportez-vous à la section "Documentation BeeGFS sur Multi-mode".

Comme les services BeeGFS doivent pouvoir démarrer sur plusieurs nœuds, le fichier de configuration pour chaque service (normalement situé à `/etc/beegfs`) Est stocké sur l'un des volumes E-Series utilisés comme cible BeeGFS pour ce service. Cela rend la configuration et les données d'un service BeeGFS accessibles à tous les nœuds qui peuvent avoir besoin d'exécuter le service.

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

Vérification de la conception

Le design de deuxième génération de la solution BeeGFS sur NetApp a été vérifié à l'aide de trois profils de configuration d'élément de base.

Les profils de configuration incluent les éléments suivants :

- Un élément de base unique, incluant la gestion BeeGFS, les métadonnées et les services de stockage.
- Des métadonnées BeeGFS plus un élément de base de stockage.
- Un élément de base BeeGFS uniquement pour le stockage.

Les éléments de base ont été reliés à deux commutateurs NVIDIA Quantum InfiniBand (MQM8700). Dix clients BeeGFS étaient également connectés aux commutateurs InfiniBand et utilisés pour exécuter des utilitaires de banc d'essai synthétiques.

La figure suivante montre la configuration BeeGFS utilisée pour valider la solution BeeGFS sur NetApp.

Répartition des fichiers BeeGFS

Les systèmes de fichiers parallèles ont notamment pour avantage de répartir les fichiers individuels sur plusieurs cibles de stockage, qui peuvent représenter des volumes sur les mêmes systèmes de stockage sous-jacents ou différents.

Dans BeeGFS, vous pouvez configurer la répartition par répertoire et par fichier pour contrôler le nombre de cibles utilisées pour chaque fichier et pour contrôler la taille chunksize (ou taille de bloc) utilisée pour chaque bande de fichier. Cette configuration permet au système de fichiers de prendre en charge différents types de charges de travail et de profils d'E/S sans avoir à reconfigurer ou à redémarrer des services. Vous pouvez appliquer les paramètres de bande à l'aide du `beegfs-ctl` Outil de ligne de commande ou avec des applications qui utilisent l'API de répartition. Pour plus d'informations, consultez la documentation BeeGFS pour "Répartition" et "API de répartition".

Pour obtenir les meilleures performances, les motifs de bande ont été ajustés tout au long des tests, et les paramètres utilisés pour chaque test sont notés.

Tests de bande passante IOR : plusieurs clients

Les tests de bande passante IOR ont utilisé OpenMPI pour exécuter des travaux parallèles du générateur d'E/S synthétique IOR (disponible à partir de "GitHub HPC") Sur l'ensemble des 10 nœuds clients à un ou plusieurs blocs de construction BeeGFS. Sauf mention contraire :

- Tous les tests ont utilisé des E/S directes avec une taille de transfert de 1MiB.
- La répartition des fichiers BeeGFS est définie sur une taille chunksize de 1 Mo et une cible par fichier.

Les paramètres suivants ont été utilisés pour IOR avec le nombre de segments ajusté afin de maintenir la taille de fichier d'agrégat à 5 Tio pour un élément de base et 40 Tio pour trois éléments de base.

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

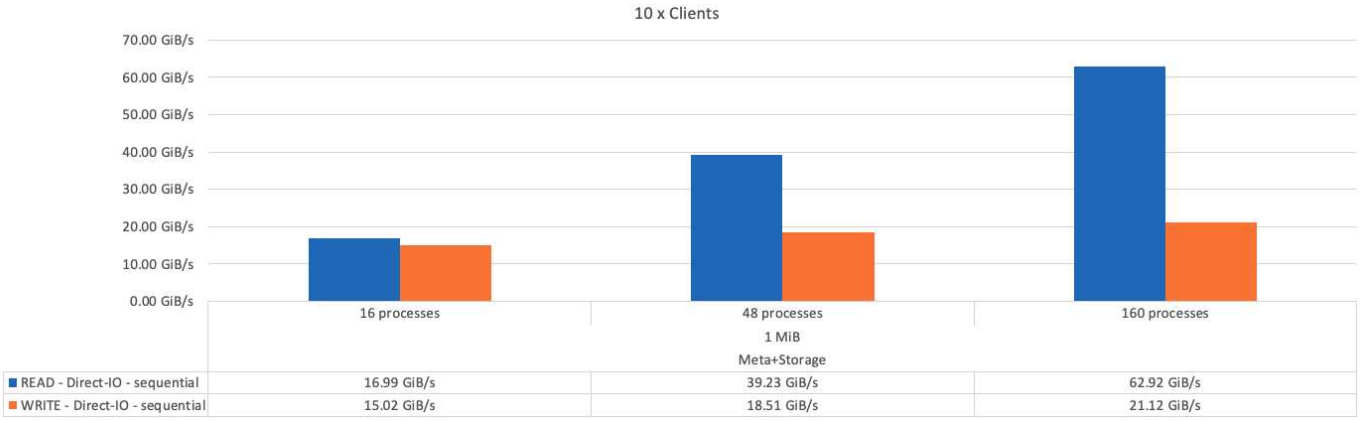
Un élément de base BeeGFS (gestion, métadonnées et stockage)

La figure suivante montre les résultats du test IOR avec un seul élément de base BeeGFS (gestion,

métadonnées et stockage).

Métadonnées BeeGFS + élément de base du stockage

La figure suivante présente les résultats du test IOR avec un seul élément de base de stockage + métadonnées BeeGFS.

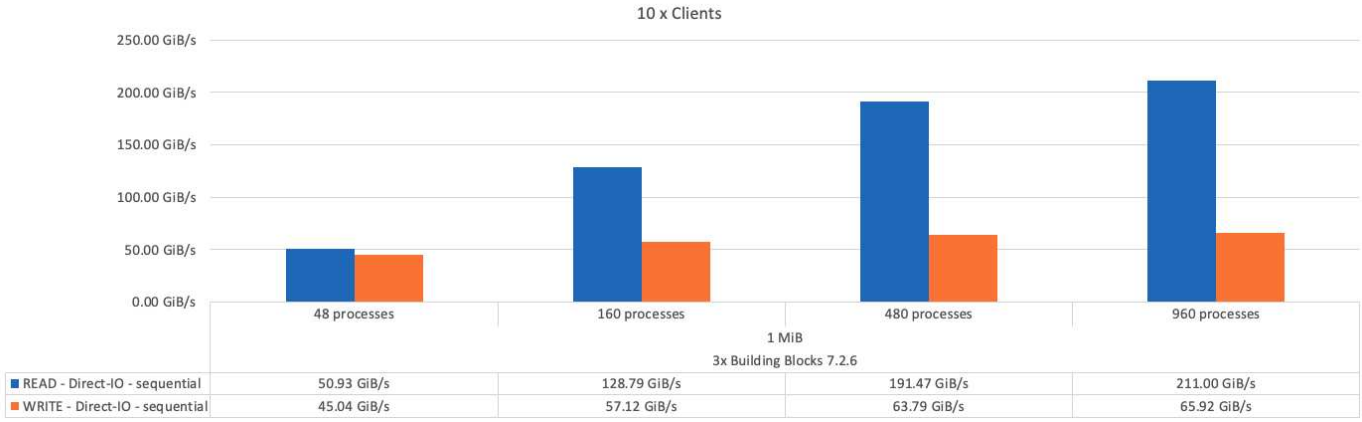


Élément de base BeeGFS uniquement pour le stockage

La figure suivante montre les résultats du test IOR avec un seul élément de base BeeGFS Storage uniquement.

Trois éléments de base BeeGFS

La figure suivante montre les résultats du test IOR avec trois éléments de base BeeGFS.



Comme on pouvait s'y attendre, la différence de performances entre l'élément de base et les métadonnées suivantes + l'élément de base du stockage est négligeable. En comparant les métadonnées + l'élément de base du stockage et un élément de base uniquement destiné au stockage, on constate une légère augmentation des performances de lecture en raison des disques supplémentaires utilisés comme cibles de stockage. Toutefois, il n'y a pas de différence significative dans les performances d'écriture. Pour améliorer les performances, vous pouvez ajouter plusieurs éléments de base pour faire évoluer les performances de manière linéaire.

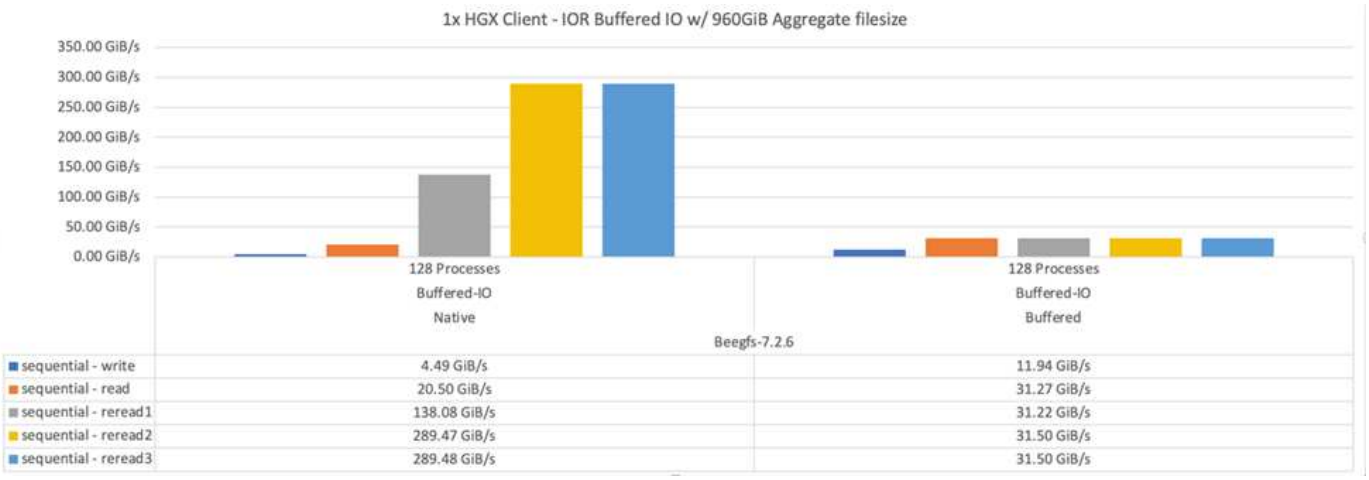
Tests de bande passante IOR : client unique

Le test de bande passante IOR a utilisé OpenMPI pour exécuter plusieurs processus IOR à l'aide d'un seul serveur GPU hautes performances afin d'explorer les performances réalisables pour un même client.

Ce test compare également le comportement et les performances de relecture de BeeGFS lorsque le client est configuré pour utiliser le cache de page du noyau Linux (`tuneFileCacheType = native`) par rapport à la valeur par défaut `buffered` réglage.

Le mode de mise en cache native utilise le cache de page du noyau Linux sur le client, ce qui permet aux opérations de relecture de provenir de la mémoire locale au lieu d'être retransmises sur le réseau.

Le diagramme suivant montre les résultats du test IOR avec trois éléments de base BeeGFS et un seul client.



La répartition BeeGFS pour ces tests a été définie sur une taille chunksize de 1 Mo avec huit cibles par fichier.

Bien que les performances d'écriture et de lecture initiale soient supérieures en mode tampon par défaut, pour

les charges de travail qui relisent plusieurs fois les mêmes données, le mode de mise en cache natif a permis d'optimiser considérablement les performances. Cette amélioration des performances de relecture est importante pour les charges de travail telles que l'apprentissage profond qui relire le même dataset plusieurs fois sur plusieurs séries de tests.

Test de performance des métadonnées

Les tests de performance des métadonnées ont utilisé l'outil MDTest (inclus dans IOR) pour mesurer la performance des métadonnées de BeeGFS. Les tests ont utilisé OpenMPI pour exécuter des travaux parallèles sur les dix nœuds clients.

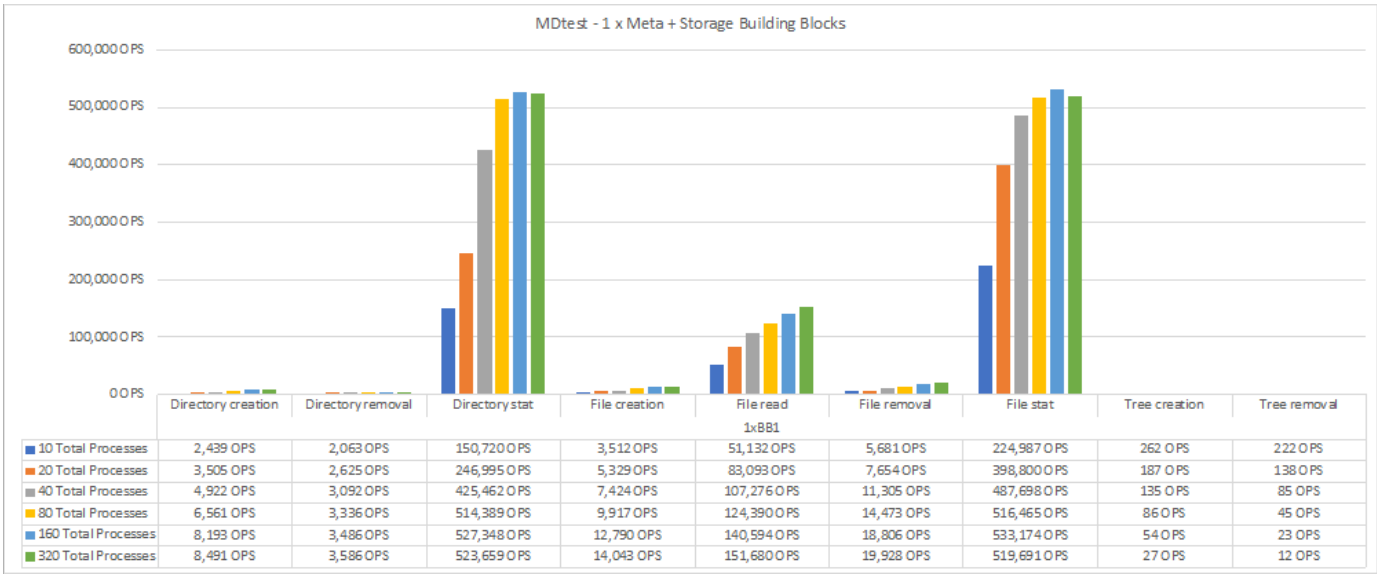
Les paramètres suivants ont été utilisés pour exécuter le test de référence avec le nombre total de processus passe de 10 à 320 par pas de 2x et avec une taille de fichier de 4 ko.

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

Les performances des métadonnées ont été mesurées en premier avec un ou deux blocs de base de stockage + métadonnées afin de montrer l'évolution des performances en ajoutant des éléments de base supplémentaires.

Un seul élément de base de métadonnées BeeGFS + stockage

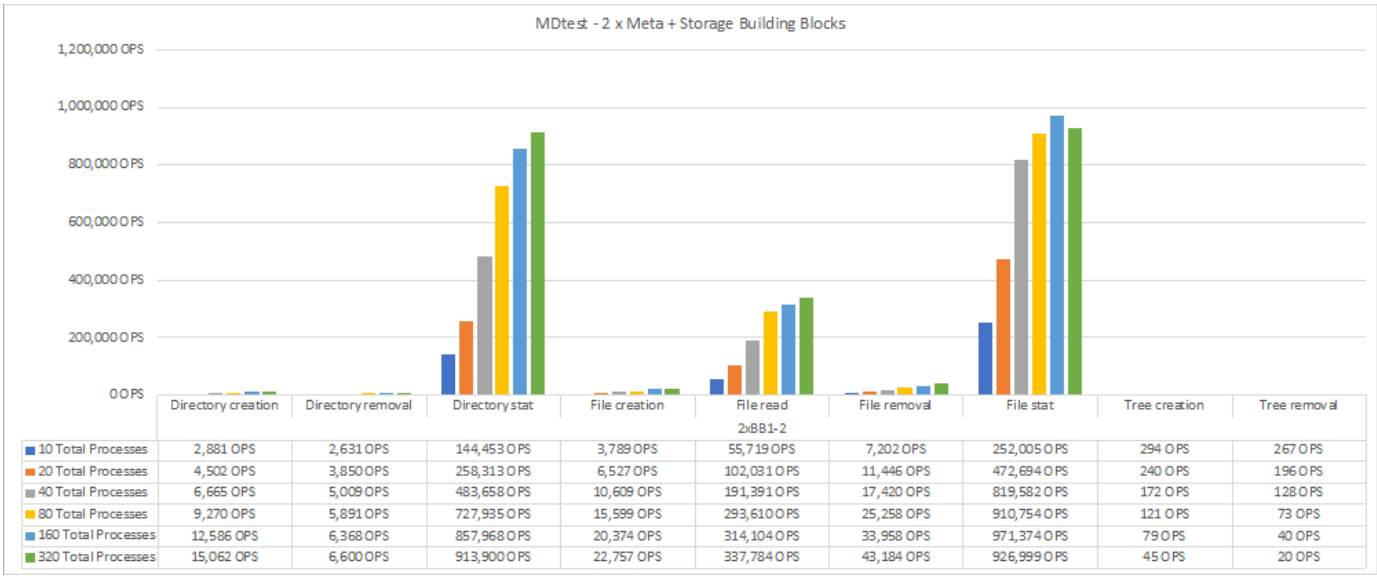
Le diagramme suivant montre les résultats MDTest avec un bloc de construction BeeGFS + stockage.



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

Deux métadonnées BeeGFS + éléments de base du stockage

Le diagramme suivant montre les résultats MDTest avec deux métadonnées BeeGFS + des modules de stockage.



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

Validation fonctionnelle

Dans le cadre de la validation de cette architecture, NetApp a effectué plusieurs tests fonctionnels :

- Défaillance d'un seul port InfiniBand client en désactivant le port de commutateur.
- Défaillance d'un seul port InfiniBand de serveur en désactivant le port du commutateur.
- Déclenchement d'une mise hors tension immédiate d'un serveur à l'aide du contrôleur BMC.
- Placement normal d'un nœud en veille et basculement de service vers un autre nœud.
- Il est normal de remettre un nœud en ligne et de renvoyer les services vers le nœud d'origine.
- Mise hors tension de l'un des commutateurs InfiniBand à l'aide de la PDU. Tous les tests ont été réalisés alors que les tests de stress étaient en cours avec le `sysSessionChecksEnabled: false` Paramètre défini sur les clients BeeGFS. Aucune erreur ni interruption des E/S n'a été observée.

> Il y a un problème connu (voir "Changement") Lorsque les connexions RDMA BeeGFS client/serveur sont interrompues de façon inattendue, soit par la perte de l'interface principale (comme défini dans la section `connInterfacesFile`) Ou un serveur BeeGFS est défaillant ; les E/S du client actif peuvent se bloquer pendant dix minutes avant de reprendre. Ce problème ne se produit pas lorsque les nœuds BeeGFS sont correctement placés en attente pour la maintenance planifiée ou si TCP est utilisé.

Validation de NVIDIA DGX SuperPOD et BasePOD

NetApp a validé une solution de stockage pour NVIDIA DGX A100 SuperPOD à l'aide d'un système de fichiers BeeGFS constitué de trois éléments de base avec les métadonnées plus le profil de configuration du stockage appliqué. L'effort de qualification a participé au test de la solution décrite par cette architecture NVA avec vingt serveurs GPU DGX A100 exécutant plusieurs bancs d'essai de stockage, d'apprentissage machine et d'apprentissage profond. Basée sur la validation établie avec le DGX A100 SuperPOD de NVIDIA, la solution BeeGFS sur NetApp a été approuvée pour les systèmes DGX SuperPOD H100, H200 et B200. Cette extension repose sur le respect des bancs d'essai et des exigences système précédemment établis et validés avec le système NVIDIA DGX A100

Pour plus d'informations, voir "NVIDIA DGX SuperPOD avec NetApp" et "NVIDIA DGX BasePOD".

Instructions de dimensionnement

La solution BeeGFS inclut des recommandations sur le dimensionnement de la performance et de la capacité qui étaient basées sur des tests de vérification.

L'objectif de l'architecture modulaire est de créer une solution simple à dimensionner en ajoutant plusieurs éléments de base pour répondre aux exigences d'un système BeeGFS particulier. À l'aide des lignes directrices ci-dessous, vous pouvez estimer la quantité et les types de blocs de construction BeeGFS qui sont nécessaires pour répondre aux exigences de votre environnement.

Notez que ces estimations sont les meilleures performances au cas par cas. Les applications de test des performances synthétiques sont écrites et utilisées pour optimiser l'utilisation des systèmes de fichiers sous-jacents d'une manière qui n'est pas forcément possible pour les applications réelles.

Dimensionnement de la performance

Le tableau suivant indique le dimensionnement de performance recommandé.

Profil de configuration	1MiB lit	Écritures 1MiB
Métadonnées + stockage	62 GiBps	21 GiBps
Stockage uniquement	64 GiBps	21 GiBps

Le dimensionnement de la capacité des métadonnées est basé sur la « règle générale » selon laquelle 500 Go de capacité suffisent pour environ 150 millions de fichiers sur BeeGFS. (Pour plus d'informations, consultez la documentation BeeGFS pour "Configuration minimale requise".)

L'utilisation de fonctions telles que les listes de contrôle d'accès et le nombre de répertoires et de fichiers par répertoire affecte également la vitesse de consommation de l'espace de métadonnées. Les estimations de capacité de stockage tiennent compte de la capacité de disque utilisable ainsi que de la surcharge RAID 6 et XFS.

Dimensionnement de la capacité pour les métadonnées + éléments de base du stockage

Le tableau suivant indique le dimensionnement de la capacité recommandé pour les métadonnées et les éléments de base du stockage.

Taille du disque (2+2 RAID 1) groupes de volumes de métadonnées	Capacité des métadonnées (nombre de fichiers)	Taille des disques (RAID 6 8+2) groupes de volumes de stockage	Capacité de stockage (contenu de fichiers)
1,92 TO	1,938,577,200	1,92 TO	51,77 TO
3,84 TO	3,880,388,400	3,84 TO	103,55 TO
7,68 TO	8,125,278,000	7,68 TO	28.74 TO
15,3 TO	17,269,854,000	15,3 TO	460.60 TO

Lors du dimensionnement des métadonnées et des éléments de base de stockage, vous pouvez réduire les coûts en utilisant des disques plus petits pour les groupes de volumes de métadonnées et les groupes de volumes de stockage.

Dimensionnement de la capacité pour les éléments de base uniquement destinés au stockage

Le tableau suivant indique la règle générale de dimensionnement de la capacité pour les éléments de base uniquement liés au stockage.

Taille des disques (RAID 6 10+2) groupes de volumes de stockage	Capacité de stockage (contenu de fichiers)
1,92 TO	59,89 TO
3,84 TO	11980 TO
7,68 TO	251,89TB
15,3 TO	58,55 TO

Les performances et la surcharge liée à la capacité de l'inclusion du service de gestion dans le premier élément de base sont minimales, sauf si le verrouillage global des fichiers est activé.

Réglage des performances

La solution BeeGFS inclut des recommandations sur le réglage de la performance qui étaient basées sur des tests de vérification.

Bien que BeeGFS fournit des performances raisonnables, NetApp a développé un ensemble de paramètres d'ajustement recommandés pour optimiser les performances. Ces paramètres prennent en compte les fonctionnalités des nœuds de bloc E-Series sous-jacents et les exigences spéciales requises pour exécuter BeeGFS dans une architecture HA à disque partagé.

L'ajustement des performances des nœuds de fichiers

Les paramètres de réglage disponibles que vous pouvez configurer sont les suivants :

1. **Paramètres système dans l'UEFI/BIOS des nœuds de fichiers.** pour optimiser les performances, nous vous recommandons de configurer les paramètres système sur le modèle de serveur que vous utilisez comme nœuds de fichiers. Vous configurez les paramètres système lorsque vous configurez vos nœuds de fichiers à l'aide de la configuration du système (UEFI/BIOS) ou des API Redfish fournies par le contrôleur de gestion de la carte mère (BMC).

   Les paramètres système varient en fonction du modèle de serveur que vous utilisez comme nœud de fichier. Les paramètres doivent être configurés manuellement en fonction du modèle de serveur utilisé. Pour savoir comment configurer les paramètres système des nœuds de fichiers Lenovo SR665 validés, reportez-vous à la section "Réglez les paramètres du système de nœud de fichiers en fonction des performances".

2. **Paramètres par défaut pour les paramètres de configuration requis.** les paramètres de configuration requis affectent la configuration des services BeeGFS et la façon dont les volumes E-Series (dispositifs de bloc) sont formatés et montés par Pacemaker. Voici les paramètres de configuration requis :

- Paramètres de configuration du service BeeGFS

  Vous pouvez remplacer les paramètres par défaut des paramètres de configuration selon vos besoins. Pour connaître les paramètres que vous pouvez ajuster en fonction de vos charges de travail ou de vos cas d'utilisation spécifiques, reportez-vous au "Paramètres de configuration du service BeeGFS".

- Le formatage de volume et les paramètres de montage sont définis sur les valeurs par défaut recommandées et ne doivent être ajustés que pour des cas d'utilisation avancés. Les valeurs par défaut sont les suivantes :
  - Optimiser le formatage du volume initial en fonction du type de cible (gestion, métadonnées ou stockage, par exemple), de la configuration RAID et de la taille du segment du volume sous-jacent.
  - Réglez la manière dont Pacemaker monte chaque volume pour vous assurer que les modifications sont immédiatement transférées vers les nœuds de blocs E-Series. Cela empêche la perte de données en cas d'échec des nœuds de fichier pour les écritures actives.

    Pour connaître les paramètres que vous pouvez ajuster en fonction de vos charges de travail ou de vos cas d'utilisation spécifiques, reportez-vous au "formatage du volume et paramètres de configuration du montage".

3. **Paramètres système du système d'exploitation Linux installé sur les nœuds de fichiers.** Lorsque vous créez l'inventaire Ansible à l'étape 4 de la section , vous pouvez remplacer les paramètres par défaut du système d'exploitation Linux "Créez l'inventaire Ansible".

   Les paramètres par défaut ont été utilisés pour valider la solution BeeGFS sur NetApp, mais vous pouvez les modifier pour s'adapter à vos workloads ou à vos utilisations spécifiques. Voici quelques exemples de paramètres système d'exploitation Linux que vous pouvez modifier :

   - Files d'attente des E/S sur les dispositifs de bloc E-Series.

     Vous pouvez configurer des files d'attente d'E/S sur les périphériques de bloc E-Series utilisés comme cibles BeeGFS pour :

     - Réglez l'algorithme de planification en fonction du type de périphérique (NVMe, HDD, etc.).
     - Augmenter le nombre de demandes en attente.
     - Réglez les tailles des demandes.
     - Optimisez le comportement de lecture anticipée.
   - Paramètres de la mémoire virtuelle.

     Vous pouvez régler les paramètres de la mémoire virtuelle pour des performances de diffusion optimales en continu.

   - Paramètres CPU.

     Vous pouvez régler le régulateur de fréquence de l'UC et d'autres configurations de l'UC pour obtenir des performances maximales.

   - Taille de la demande de lecture.

     Vous pouvez augmenter la taille maximale des demandes de lecture pour les applications HCA NVIDIA.

Réglage des performances des nœuds en mode bloc

En fonction des profils de configuration appliqués à un élément de base BeeGFS particulier, les groupes de volumes configurés sur les nœuds de blocs changent légèrement. Par exemple, avec un nœud de bloc EF600 de 24 disques :

- Pour un seul élément de base, y compris la gestion BeeGFS, les métadonnées et les services de stockage :
  - 1 groupe de volumes RAID 10 2+2 pour la gestion BeeGFS et les services de métadonnées
  - 2 groupes de volumes RAID 6 8+2 pour les services de stockage BeeGFS
- Pour un élément de base de métadonnées + de stockage BeeGFS :
  - 1 groupe de volumes RAID 10 2+2 pour les services de métadonnées BeeGFS
  - 2 groupes de volumes RAID 6 8+2 pour les services de stockage BeeGFS
- Pour l'élément de base de stockage BeeGFS uniquement :
  - 2 groupes de volumes RAID 6 10+2 pour les services de stockage BeeGFS

> Comme BeeGFS a besoin d'un espace de stockage considérable pour la gestion et les métadonnées par rapport au stockage, une seule option consiste à utiliser des disques plus petits pour les groupes de volumes RAID 10. Les lecteurs plus petits doivent être insérés dans les emplacements de lecteur les plus extérieurs. Pour plus d'informations, reportez-vous à la section "instructions de déploiement".

Tous ces paramètres sont configurés par le déploiement Ansible, et plusieurs autres paramètres sont généralement recommandés pour optimiser les performances/comportements :

- Ajustement de la taille du bloc de cache global à 32Kio et ajustement de la vidage du cache à la demande à 80 %.
- Désactivation de l'équilibrage automatique (en veillant à ce que les attributions de volume du contrôleur restent telles que prévues).
- Activation de la mise en cache de lecture et désactivation de la mise en cache de lecture anticipée
- Activation de la mise en cache d'écriture avec la mise en miroir et demande de sauvegarde sur batterie, les caches sont donc conservés suite à la panne d'un contrôleur de nœud bloc.
- Spécification de l'ordre dans lequel les disques sont affectés aux groupes de volumes, en équilibrant les E/S entre les canaux de disque disponibles.

Élément de base haute capacité

La solution BeeGFS est conçue de façon très performante. Les clients recherchant des cas d'utilisation de grande capacité doivent observer les variations des caractéristiques de conception et de performances décrites ici.

Configuration matérielle et logicielle

Les configurations matérielles et logicielles de l'élément de base haute capacité sont standard, mais les contrôleurs EF600 doivent être remplacés par des contrôleurs EF300, avec une option qui permet de connecter entre 1 et 7 tiroirs d'extension IOM avec 60 disques chacun pour chaque baie de stockage, un total de 2 à 14 tiroirs d'extension par module.

Les clients qui déploient un design d'éléments de base haute capacité n'utilisent probablement que la configuration de type élément de base, qui comprend la gestion BeeGFS, les métadonnées et les services de stockage pour chaque nœud. Pour garantir une rentabilité accrue, les nœuds de stockage haute capacité doivent provisionner des volumes de métadonnées sur les disques NVMe du boîtier de contrôleur EF300 et provisionner les volumes de stockage sur les disques NL-SAS des tiroirs d'extension.

[]

Instructions de dimensionnement

Ces recommandations de dimensionnement supposent que les blocs de base haute capacité sont configurés avec un groupe de volumes SSD NVMe 2+2 pour les métadonnées dans le boîtier EF300 de base et six groupes de volumes NL-SAS 8+2 par plateau d'extension IOM pour le stockage.

Taille du disque (disques durs haute capacité)	Capacité par BB (1 plateau)	Capacité par BB (2 plateaux)	Capacité par BB (3 plateaux)	Capacité par BB (4 plateaux)
4 TO	43TB	878 TO	1317 TO	1756 TO
8 TO	878 TO	1756 TO	2634 TO	3512 TO
10 TO	1097 TO	2195 TO	3292 TO	4390 TO
12 To	1317 TO	2634 TO	3951 TO	5268 TO
16 TO	1756 TO	3512 TO	5268 TO	7024 TO
18 TO	1975 TO	3951 TO	5927 TO	7902 TO

Informations sur le copyright

Informations sur les marques commerciales