Red Hat Enterprise Linux 9

Configuration des systèmes de fichiers GFS2

Planification, administration, dépannage et configuration des systèmes de fichiers GFS2 dans un cluster à haute disponibilité

Last Updated: 2024-01-18

Red Hat Enterprise Linux 9 Configuration des systèmes de fichiers GFS2

Planification, administration, dépannage et configuration des systèmes de fichiers GFS2 dans un cluster à haute disponibilité

Notice légale

Résumé

Red Hat Enterprise Linux (RHEL) Resilient Storage Add-On fournit le Red Hat Global File System 2 (GFS2), un système de fichiers en cluster qui gère la cohérence entre plusieurs nœuds partageant un périphérique de bloc commun. Ce titre fournit des informations sur la planification du déploiement d'un système de fichiers GFS2 ainsi que des procédures de configuration, de dépannage et de réglage des systèmes de fichiers GFS2.

Table des matières

RENDRE L'OPEN SOURCE PLUS INCLUSIF

Red Hat s'engage à remplacer les termes problématiques dans son code, sa documentation et ses propriétés Web. Nous commençons par ces quatre termes : master, slave, blacklist et whitelist. En raison de l'ampleur de cette entreprise, ces changements seront mis en œuvre progressivement au cours de plusieurs versions à venir. Pour plus de détails, voir le message de notre directeur technique Chris Wright.

FOURNIR UN RETOUR D'INFORMATION SUR LA DOCUMENTATION DE RED HAT

Nous apprécions vos commentaires sur notre documentation. Faites-nous savoir comment nous pouvons l'améliorer.

Soumettre des commentaires sur des passages spécifiques

1. Consultez la documentation au format **Multi-page HTML** et assurez-vous que le bouton **Feedback** apparaît dans le coin supérieur droit après le chargement complet de la page.
2. Utilisez votre curseur pour mettre en évidence la partie du texte que vous souhaitez commenter.
3. Cliquez sur le bouton **Add Feedback** qui apparaît près du texte en surbrillance.
4. Ajoutez vos commentaires et cliquez sur **Submit**.

Soumettre des commentaires via Bugzilla (compte requis)

1. Connectez-vous au site Web de Bugzilla.
2. Sélectionnez la version correcte dans le menu **Version**.
3. Saisissez un titre descriptif dans le champ **Summary**.
4. Saisissez votre suggestion d'amélioration dans le champ **Description**. Incluez des liens vers les parties pertinentes de la documentation.
5. Cliquez sur **Submit Bug**.

CHAPITRE 1. PLANIFIER LE DÉPLOIEMENT D'UN SYSTÈME DE FICHIERS GFS2

Le système de fichiers Red Hat Global File System 2 (GFS2) est un système de fichiers en grappe symétrique de 64 bits qui fournit un espace de noms partagé et gère la cohérence entre plusieurs nœuds partageant un périphérique de bloc commun. Un système de fichiers GFS2 est destiné à fournir un ensemble de fonctionnalités aussi proche que possible d'un système de fichiers local, tout en appliquant une cohérence de cluster complète entre les nœuds. Pour ce faire, les nœuds utilisent un système de verrouillage à l'échelle de la grappe pour les ressources du système de fichiers. Ce schéma de verrouillage utilise des protocoles de communication tels que TCP/IP pour échanger des informations de verrouillage.

Dans certains cas, l'API du système de fichiers Linux ne permet pas de rendre totalement transparente la nature groupée de GFS2 ; par exemple, les programmes utilisant des verrous POSIX dans GFS2 doivent éviter d'utiliser la fonction **GETLK** car, dans un environnement groupé, l'ID du processus peut être celui d'un nœud différent dans le groupe. Dans la plupart des cas, cependant, la fonctionnalité d'un système de fichiers GFS2 est identique à celle d'un système de fichiers local.

Le module complémentaire de stockage résilient de Red Hat Enterprise Linux (RHEL) fournit GFS2 et dépend du module complémentaire de haute disponibilité de RHEL pour fournir la gestion de cluster requise par GFS2.

Le module du noyau **gfs2.ko** implémente le système de fichiers GFS2 et est chargé sur les nœuds du cluster GFS2.

Pour obtenir les meilleures performances de GFS2, il est important de prendre en compte les considérations de performance qui découlent de la conception sous-jacente. Tout comme un système de fichiers local, GFS2 s'appuie sur le cache de pages pour améliorer les performances grâce à la mise en cache locale des données fréquemment utilisées. Afin de maintenir la cohérence entre les nœuds du cluster, le contrôle du cache est assuré par la machine d'état *glock*.

IMPORTANT

Assurez-vous que votre déploiement de Red Hat High Availability Add-On répond à vos besoins et peut être pris en charge. Consultez un représentant autorisé de Red Hat pour vérifier votre configuration avant le déploiement.

1.1. FORMAT DU SYSTÈME DE FICHIERS GFS2 VERSION 1802

À partir de Red Hat Enterprise Linux 9, les systèmes de fichiers GFS2 sont créés avec le format version 1802.

La version 1802 du format permet les fonctionnalités suivantes :

- Les attributs étendus de l'espace de noms **trusted** ("trusted.* xattrs") sont reconnus par **gfs2** et **gfs2-utils**.
- L'option **rgrplvb** est active par défaut. Elle permet à **gfs2** d'attacher des données de groupe de ressources mises à jour aux demandes de verrouillage DLM, de sorte que le nœud qui acquiert le verrouillage n'a pas besoin de mettre à jour les informations de groupe de ressources à partir du disque. Cela améliore les performances dans certains cas.

Les systèmes de fichiers créés avec la nouvelle version du format ne pourront pas être montés sous les versions antérieures de RHEL et les anciennes versions de l'utilitaire **fsck.gfs2** ne pourront pas les vérifier.

Les utilisateurs peuvent créer un système de fichiers avec l'ancienne version du format en exécutant la commande **mkfs.gfs2** avec l'option **-o format=1801**.

Les utilisateurs peuvent mettre à jour la version du format d'un ancien système de fichiers fonctionnant sur un système de fichiers non monté **tunegfs2 -r 1802 *device*** sur un système de fichiers non monté. La rétrogradation de la version du format n'est pas prise en charge.

1.2. PARAMÈTRES CLÉS DE GFS2 À DÉTERMINER

Avant d'installer et de configurer un système de fichiers GFS2, vous devez tenir compte d'un certain nombre de paramètres clés.

Nœuds GFS2

Déterminez les nœuds de la grappe qui monteront les systèmes de fichiers GFS2.

Nombre de systèmes de fichiers

Déterminez le nombre de systèmes de fichiers GFS2 à créer initialement. D'autres systèmes de fichiers peuvent être ajoutés ultérieurement.

Nom du système de fichiers

Chaque système de fichiers GFS2 doit avoir un nom unique. Ce nom est généralement identique au nom du volume logique LVM et est utilisé comme nom de table de verrouillage DLM lorsqu'un système de fichiers GFS2 est monté. Par exemple, ce guide utilise les noms de systèmes de fichiers **mydata1** et **mydata2** dans certains exemples de procédures.

Journaux

Déterminez le nombre de journaux pour vos systèmes de fichiers GFS2. GFS2 nécessite un journal pour chaque nœud de la grappe qui doit monter le système de fichiers. Par exemple, si vous avez une grappe de 16 nœuds mais que vous ne devez monter le système de fichiers qu'à partir de deux nœuds, vous n'avez besoin que de deux journaux. GFS2 vous permet d'ajouter dynamiquement des journaux à l'aide de l'utilitaire **gfs2_jadd** lorsque des serveurs supplémentaires montent un système de fichiers.

Périphériques de stockage et partitions

Déterminez les périphériques de stockage et les partitions à utiliser pour créer des volumes logiques (à l'aide de **lvmlockd**) dans les systèmes de fichiers.

Protocole temporel

Assurez-vous que les horloges des nœuds GFS2 sont synchronisées. Il est recommandé d'utiliser le Precision Time Protocol (PTP) ou, si cela est nécessaire pour votre configuration, le logiciel Network Time Protocol (NTP) fourni avec votre distribution Red Hat Enterprise Linux.

Les horloges système des nœuds GFS2 doivent se situer à quelques minutes d'intervalle pour éviter une mise à jour inutile de l'horodatage des inodes. La mise à jour inutile de l'horodatage de l'inode a un impact important sur les performances du cluster.

NOTE

Vous pouvez rencontrer des problèmes de performance avec GFS2 lorsque de nombreuses opérations de création et de suppression sont effectuées simultanément dans le même répertoire à partir de plusieurs nœuds. Si cela pose des problèmes de performance dans votre système, vous devriez localiser la création et la suppression de fichiers par un nœud dans des répertoires spécifiques à ce nœud, dans la mesure du possible.

1.3. CONSIDÉRATIONS RELATIVES À LA PRISE EN CHARGE DE GFS2

Pour pouvoir bénéficier de l'assistance de Red Hat pour un cluster exécutant un système de fichiers GFS2, vous devez prendre en compte les politiques d'assistance pour les systèmes de fichiers GFS2.

1.3.1. Taille maximale du système de fichiers et du cluster

Le tableau suivant résume la taille maximale actuelle du système de fichiers et le nombre de nœuds pris en charge par GFS2.

Tableau 1.1. Limites du support GFS2

Paramètres	Maximum
Nombre de nœuds	16 (x86, Power8 sur PowerVM) 4 (s390x sous z/VM)
Taille du système de fichiers	100TB sur toutes les architectures supportées

GFS2 est basé sur une architecture 64 bits, qui peut théoriquement accueillir un système de fichiers de 8 EB. Si votre système nécessite des systèmes de fichiers GFS2 plus importants que ceux actuellement pris en charge, contactez votre représentant Red Hat.

Lorsque vous déterminez la taille de votre système de fichiers, vous devez tenir compte de vos besoins en matière de récupération. L'exécution de la commande **fsck.gfs2** sur un système de fichiers très volumineux peut prendre beaucoup de temps et consommer une grande quantité de mémoire. En outre, en cas de défaillance d'un disque ou d'un sous-système de disque, le temps de récupération est limité par la vitesse de votre support de sauvegarde. Pour plus d'informations sur la quantité de mémoire nécessaire à la commande **fsck.gfs2**, voir Détermination de la mémoire nécessaire à l'exécution de fsck.gfs2.

1.3.2. Taille minimale de la grappe

Bien qu'un système de fichiers GFS2 puisse être implémenté dans un système autonome ou dans le cadre d'une configuration en grappe, Red Hat ne prend pas en charge l'utilisation de GFS2 en tant que système de fichiers à nœud unique, à l'exception des cas suivants :

- Red Hat prend en charge les systèmes de fichiers GFS2 à nœud unique pour le montage d'instantanés de systèmes de fichiers en grappe qui pourraient être nécessaires, par exemple, à des fins de sauvegarde.

- Un cluster à nœud unique montant des systèmes de fichiers GFS2 (qui utilise DLM) est pris en charge en tant que nœud de reprise après sinistre (DR) sur un site secondaire. Cette exception n'est valable qu'à des fins de reprise après sinistre et non pour transférer la charge de travail du cluster principal vers le site secondaire.
  Par exemple, la copie des données du système de fichiers monté sur le site secondaire alors que le site principal est hors ligne est prise en charge. Toutefois, la migration d'une charge de travail du site principal directement vers un site secondaire à nœud unique n'est pas prise en charge. Si la charge de travail complète doit être migrée vers le site secondaire à nœud unique, le site secondaire doit être de la même taille que le site primaire.

  Red Hat recommande que lorsque vous montez un système de fichiers GFS2 dans un cluster à un seul nœud, vous spécifiez l'option de montage **errors=panic** afin que le cluster à un seul nœud panique lorsqu'un retrait GFS2 se produit puisque le cluster à un seul nœud ne sera pas en mesure de se clôturer lui-même lorsqu'il rencontrera des erreurs de système de fichiers.

Red Hat prend en charge un certain nombre de systèmes de fichiers haute performance à nœud unique qui sont optimisés pour un nœud unique et qui ont donc généralement moins de frais généraux qu'un système de fichiers en grappe. Red Hat recommande d'utiliser ces systèmes de fichiers de préférence à GFS2 dans les cas où un seul nœud doit monter le système de fichiers. Pour plus d'informations sur les systèmes de fichiers pris en charge par Red Hat Enterprise Linux 9, reportez-vous à Gestion des systèmes de fichiers.

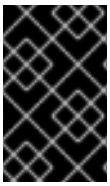
1.3.3. Considérations sur le stockage partagé

Bien qu'un système de fichiers GFS2 puisse être utilisé en dehors de LVM, Red Hat ne prend en charge que les systèmes de fichiers GFS2 créés sur un volume logique LVM partagé.

Lorsque vous configurez un système de fichiers GFS2 en tant que système de fichiers en grappe, vous devez vous assurer que tous les nœuds de la grappe ont accès au stockage partagé. Les configurations asymétriques dans lesquelles certains nœuds ont accès au stockage partagé et d'autres non ne sont pas prises en charge. Il n'est pas nécessaire que tous les nœuds montent le système de fichiers GFS2 lui-même.

1.4. CONSIDÉRATIONS SUR LE FORMATAGE GFS2

Pour formater votre système de fichiers GFS2 afin d'en optimiser les performances, vous devez tenir compte de ces recommandations.



IMPORTANT

Assurez-vous que votre déploiement de Red Hat High Availability Add-On répond à vos besoins et peut être pris en charge. Consultez un représentant autorisé de Red Hat pour vérifier votre configuration avant le déploiement.

Taille du système de fichiers : Plus c'est petit, mieux c'est
GFS2 est basé sur une architecture 64 bits, qui peut théoriquement accueillir un système de fichiers de 8 EB. Cependant, la taille maximale actuellement supportée par un système de fichiers GFS2 pour un matériel 64 bits est de 100 To.

Notez que même si les systèmes de fichiers GFS2 de grande taille sont possibles, cela ne signifie pas qu'ils sont recommandés. La règle de base avec GFS2 est que plus c'est petit, mieux c'est : il est préférable d'avoir 10 systèmes de fichiers de 1 To plutôt qu'un système de fichiers de 10 To.

Il y a plusieurs raisons pour lesquelles vous devriez garder vos systèmes de fichiers GFS2 de petite taille :

- La sauvegarde de chaque système de fichiers prend moins de temps.
- Il faut moins de temps pour vérifier le système de fichiers à l'aide de la commande **fsck.gfs2**.
- Moins de mémoire est nécessaire si vous devez vérifier le système de fichiers avec la commande **fsck.gfs2**.

En outre, la réduction du nombre de groupes de ressources à gérer se traduit par de meilleures performances.

Bien entendu, si votre système de fichiers GFS2 est trop petit, vous risquez de manquer d'espace, ce qui n'est pas sans conséquences. Vous devez prendre en compte vos propres cas d'utilisation avant de décider de la taille du système.

Taille des blocs : Les blocs par défaut (4K) sont privilégiés

La commande **mkfs.gfs2** tente d'estimer une taille de bloc optimale basée sur la topologie du périphérique. En général, les blocs de 4K sont la taille de bloc préférée car 4K est la taille de page par défaut (mémoire) pour Red Hat Enterprise Linux. Contrairement à d'autres systèmes de fichiers, GFS2 effectue la plupart de ses opérations en utilisant des tampons de noyau de 4K. Si la taille de votre bloc est de 4K, le noyau doit effectuer moins de travail pour manipuler les tampons.

Il est recommandé d'utiliser la taille de bloc par défaut, qui devrait permettre d'obtenir les meilleures performances. L'utilisation d'une taille de bloc différente n'est nécessaire que si vous avez besoin de stocker efficacement de nombreux petits fichiers.

Taille du journal : La valeur par défaut (128 Mo) est généralement optimale

Lorsque vous exécutez la commande **mkfs.gfs2** pour créer un système de fichiers GFS2, vous pouvez spécifier la taille des journaux. Si vous n'indiquez pas de taille, la valeur par défaut est de 128 Mo, ce qui devrait être optimal pour la plupart des applications.

Certains administrateurs de système pourraient penser que 128 Mo est excessif et être tentés de réduire la taille du journal au minimum de 8 Mo ou à 32 Mo, ce qui est plus prudent. Bien que cette solution puisse fonctionner, elle peut avoir un impact important sur les performances. Comme de nombreux systèmes de fichiers avec journalisation, chaque fois que GFS2 écrit des métadonnées, celles-ci sont validées dans le journal avant d'être mises en place. Cela garantit que si le système tombe en panne ou perd de l'énergie, vous récupérerez toutes les métadonnées lorsque le journal sera automatiquement rejoué au moment du montage. Cependant, il ne faut pas beaucoup d'activité au système de fichiers pour remplir un journal de 8 Mo, et lorsque le journal est plein, les performances ralentissent car GFS2 doit attendre les écritures sur le stockage.

Il est généralement recommandé d'utiliser la taille de journal par défaut de 128 Mo. Si votre système de fichiers est très petit (par exemple, 5 Go), l'utilisation d'un journal de 128 Mo peut s'avérer peu pratique. Si vous disposez d'un système de fichiers plus important et que vous pouvez vous permettre d'utiliser l'espace nécessaire, l'utilisation de journaux de 256 Mo peut améliorer les performances.

Taille et nombre des groupes de ressources

Lorsqu'un système de fichiers GFS2 est créé à l'aide de la commande **mkfs.gfs2**, il divise le stockage en tranches uniformes appelées groupes de ressources. Il tente d'estimer une taille optimale pour les groupes de ressources (entre 32 Mo et 2 Go). Vous pouvez remplacer la valeur par défaut avec l'option **-r** de la commande **mkfs.gfs2**.

La taille optimale du groupe de ressources dépend de l'utilisation que vous ferez du système de fichiers. Tenez compte du degré de saturation du système et de la présence ou non d'une fragmentation importante.

Il est conseillé d'expérimenter différentes tailles de groupes de ressources afin de déterminer celle qui permet d'obtenir des performances optimales. La meilleure pratique consiste à expérimenter avec un cluster de test avant de déployer GFS2 en production complète.

Si votre système de fichiers comporte trop de groupes de ressources, chacun d'entre eux étant trop petit, les allocations de blocs peuvent perdre trop de temps à rechercher un bloc libre dans des dizaines de milliers de groupes de ressources. Plus votre système de fichiers est saturé, plus le nombre de groupes de ressources à rechercher est élevé, et chacun d'entre eux nécessite un verrou à l'échelle de la grappe. Les performances s'en trouvent ralenties.

Toutefois, si votre système de fichiers comporte trop peu de groupes de ressources, chacun d'entre eux étant trop grand, les allocations de blocs risquent de se disputer plus souvent le même verrou de groupe de ressources, ce qui a également un impact sur les performances. Par exemple, si vous avez un système de fichiers de 10 Go divisé en cinq groupes de ressources de 2 Go, les nœuds de votre cluster se disputeront ces cinq groupes de ressources plus souvent que si le même système de fichiers était divisé

en 320 groupes de ressources de 32 Mo. Le problème est exacerbé si votre système de fichiers est presque plein, car chaque allocation de bloc peut devoir parcourir plusieurs groupes de ressources avant d'en trouver un avec un bloc libre. GFS2 tente d'atténuer ce problème de deux manières :

- Tout d'abord, lorsqu'un groupe de ressources est complètement plein, il s'en souvient et essaie d'éviter de le vérifier pour de futures allocations jusqu'à ce qu'un bloc soit libéré. Si vous ne supprimez jamais de fichiers, la contention sera moins importante. Toutefois, si votre application supprime constamment des blocs et en alloue de nouveaux sur un système de fichiers qui est en grande partie plein, la contention sera très élevée, ce qui aura un impact important sur les performances.

- Deuxièmement, lorsque de nouveaux blocs sont ajoutés à un fichier existant (par exemple, par ajout), GFS2 tente de regrouper les nouveaux blocs dans le même groupe de ressources que le fichier. Cela permet d'améliorer les performances : sur un disque en rotation, les opérations de recherche prennent moins de temps lorsqu'elles sont physiquement proches les unes des autres.

Le pire scénario est celui d'un répertoire central dans lequel tous les nœuds créent des fichiers, car tous les nœuds se battent constamment pour verrouiller le même groupe de ressources.

1.5. CONSIDÉRATIONS RELATIVES À GFS2 DANS UN CLUSTER

Lorsque vous déterminez le nombre de nœuds que votre système contiendra, notez qu'il existe un compromis entre la haute disponibilité et les performances. Avec un plus grand nombre de nœuds, il devient de plus en plus difficile de faire évoluer les charges de travail. Pour cette raison, Red Hat ne prend pas en charge l'utilisation de GFS2 pour les déploiements de systèmes de fichiers en cluster de plus de 16 nœuds.

Le déploiement d'un système de fichiers en grappe ne remplace pas le déploiement d'un nœud unique. Red Hat recommande de prévoir une période d'environ 8 à 12 semaines de tests sur les nouvelles installations afin de tester le système et de s'assurer qu'il fonctionne au niveau de performance requis. Au cours de cette période, tout problème de performance ou de fonctionnement peut être résolu et toute question doit être adressée à l'équipe d'assistance de Red Hat.

Red Hat recommande aux clients qui envisagent de déployer des clusters de faire réviser leurs configurations par le support Red Hat avant le déploiement afin d'éviter tout problème de support ultérieur.

1.6. CONSIDÉRATIONS SUR LE MATÉRIEL

Tenez compte des considérations matérielles suivantes lors du déploiement d'un système de fichiers GFS2.

- Utiliser des options de stockage de meilleure qualité
  GFS2 peut fonctionner sur des options de stockage partagé moins coûteuses, telles que iSCSI ou Fibre Channel over Ethernet (FCoE), mais vous obtiendrez de meilleures performances si vous achetez un stockage de meilleure qualité avec une plus grande capacité de mise en cache. Red Hat effectue la plupart des tests de qualité, d'intégrité et de performance sur un stockage SAN avec une interconnexion Fibre Channel. En règle générale, il est toujours préférable de déployer quelque chose qui a été testé en premier.

- Tester l'équipement du réseau avant de le déployer
  Un équipement réseau de meilleure qualité et plus rapide permet aux communications des clusters et à GFS2 de fonctionner plus rapidement et avec une meilleure fiabilité. Toutefois, il n'est pas nécessaire d'acheter le matériel le plus cher. Certains des commutateurs réseau les

plus chers ont des difficultés à transmettre les paquets multicast, qui sont utilisés pour transmettre les verrous **fcntl** (flocks), alors que les commutateurs réseau de base moins chers sont parfois plus rapides et plus fiables. Red Hat recommande d'essayer le matériel avant de le déployer en production complète.

CHAPITRE 2. RECOMMANDATIONS POUR L'UTILISATION DE GFS2

Lors du déploiement d'un système de fichiers GFS2, il existe un certain nombre de recommandations générales à prendre en compte.

2.1. CONFIGURATION DES MISES À JOUR DE ATIME

Chaque inode de fichier et de répertoire est associé à trois horodatages :

- **ctime** - La dernière fois que l'état de l'inode a été modifié
- **mtime** - Dernière modification des données du fichier (ou du répertoire)
- **atime** - Dernier accès aux données du fichier (ou du répertoire)

Si les mises à jour de **atime** sont activées, comme c'est le cas par défaut sur GFS2 et d'autres systèmes de fichiers Linux, chaque fois qu'un fichier est lu, son inode doit être mis à jour.

Étant donné que peu d'applications utilisent les informations fournies par **atime**, ces mises à jour peuvent nécessiter une quantité importante de trafic d'écriture et de verrouillage de fichiers inutiles. Ce trafic peut dégrader les performances ; il peut donc être préférable de désactiver les mises à jour de **atime** ou d'en réduire la fréquence.

Les méthodes suivantes permettent de réduire les effets de la mise à jour de **atime**:

- Montage avec **relatime** (atime relatif), qui met à jour **atime** si la précédente mise à jour **atime** est plus ancienne que la mise à jour **mtime** ou **ctime**. Il s'agit de l'option de montage par défaut pour les systèmes de fichiers GFS2.
- Monter avec **noatime** ou **nodiratime**. Le montage avec **noatime** désactive les mises à jour **atime** pour les fichiers et les répertoires de ce système de fichiers, tandis que le montage avec **nodiratime** désactive les mises à jour **atime** uniquement pour les répertoires de ce système de fichiers. Il est généralement recommandé de monter les systèmes de fichiers GFS2 avec l'option de montage **noatime** ou **nodiratime** dans la mesure du possible, avec une préférence pour **noatime** lorsque l'application le permet. Pour plus d'informations sur l'effet de ces arguments sur les performances du système de fichiers GFS2, voir Verrouillage des nœuds GFS2.

Utilisez la commande suivante pour monter un système de fichiers GFS2 avec l'option de montage **noatime** Linux.

```
mount BlockDevice MountPoint -o noatime
```

BlockDevice

Spécifie le périphérique de bloc où réside le système de fichiers GFS2.

MountPoint

Spécifie le répertoire dans lequel le système de fichiers GFS2 doit être monté.

Dans cet exemple, le système de fichiers GFS2 réside sur **/dev/vg01/lvol0** et est monté sur le répertoire **/mygfs2**, les mises à jour de **atime** étant désactivées.

```
# mount /dev/vg01/lvol0 /mygfs2 -o noatime
```

2.2. OPTIONS DE RÉGLAGE DU VFS : RECHERCHE ET EXPÉRIMENTATION

Comme tous les systèmes de fichiers Linux, GFS2 repose sur une couche appelée système de fichiers virtuels (VFS). Le VFS fournit de bonnes valeurs par défaut pour les paramètres de cache pour la plupart des charges de travail et ne devrait pas avoir besoin d'être modifié dans la plupart des cas. Toutefois, si vous avez une charge de travail qui ne fonctionne pas efficacement (par exemple, le cache est trop grand ou trop petit), vous pouvez améliorer les performances en utilisant la commande **sysctl**(8) pour ajuster les valeurs des fichiers **sysctl** dans le répertoire **/proc/sys/vm**. La documentation de ces fichiers se trouve dans l'arborescence des sources du noyau **Documentation/sysctl/vm.txt**.

Par exemple, les valeurs de **dirty_background_ratio** et **vfs_cache_pressure** peuvent être ajustées en fonction de votre situation. Pour obtenir les valeurs actuelles, utilisez les commandes suivantes :

```
# sysctl -n vm.dirty_background_ratio
# sysctl -n vm.vfs_cache_pressure
```

Les commandes suivantes permettent d'ajuster les valeurs :

```
# sysctl -w vm.dirty_background_ratio=20
# sysctl -w vm.vfs_cache_pressure=500
```

Vous pouvez modifier de façon permanente les valeurs de ces paramètres en éditant le fichier **/etc/sysctl.conf**.

Pour trouver les valeurs optimales pour vos cas d'utilisation, recherchez les différentes options VFS et expérimentez sur un cluster de test avant de le déployer en production.

2.3. SELINUX SUR GFS2

L'utilisation de Security Enhanced Linux (SELinux) avec GFS2 entraîne une légère pénalité en termes de performances. Pour éviter cette surcharge, vous pouvez choisir de ne pas utiliser SELinux avec GFS2, même sur un système où SELinux est en mode d'application. Lors du montage d'un système de fichiers GFS2, vous pouvez vous assurer que SELinux n'essaiera pas de lire l'élément **seclabel** sur chaque objet du système de fichiers en utilisant l'une des options **context** décrites dans la page de manuel **mount**(8) ; SELinux supposera que tout le contenu du système de fichiers est étiqueté avec l'élément **seclabel** fourni dans les options de montage **context**. Cela accélérera également le traitement en évitant une autre lecture sur disque du bloc d'attributs étendu qui pourrait contenir des éléments **seclabel**.

Par exemple, sur un système doté de SELinux en mode renforcé, vous pouvez utiliser la commande **mount** suivante pour monter le système de fichiers GFS2 s'il doit contenir du contenu Apache. Cette étiquette s'applique à l'ensemble du système de fichiers ; elle reste en mémoire et n'est pas écrite sur le disque.

```
# mount -t gfs2 -o context=system_u:object_r:httpd_sys_content_t:s0
/dev/mapper/xyz/mnt/gfs2
```

Si vous n'êtes pas sûr que le système de fichiers contiendra du contenu Apache, vous pouvez utiliser les étiquettes **public_content_rw_t** ou **public_content_t**, ou vous pouvez définir une nouvelle étiquette et définir une politique autour d'elle.

Notez que dans un cluster Pacemaker, vous devez toujours utiliser Pacemaker pour gérer un système de fichiers GFS2. Vous pouvez spécifier les options de montage lorsque vous créez une ressource de système de fichiers GFS2.

2.4. CONFIGURATION DE NFS SUR GFS2

En raison de la complexité accrue du sous-système de verrouillage GFS2 et de sa nature en grappe, la configuration de NFS sur GFS2 nécessite de nombreuses précautions.

AVERTISSEMENT

Si le système de fichiers GFS2 est exporté par NFS, vous devez monter le système de fichiers avec l'option **localflocks**. Étant donné que l'utilisation de l'option **localflocks** vous empêche d'accéder en toute sécurité au système de fichiers GFS2 à partir de plusieurs emplacements et qu'il n'est pas viable d'exporter GFS2 à partir de plusieurs nœuds simultanément, le support exige que le système de fichiers GFS2 soit monté sur un seul nœud à la fois lors de l'utilisation de cette configuration. L'effet recherché est de forcer les verrous POSIX de chaque serveur à être locaux : non groupés, indépendants les uns des autres. En effet, un certain nombre de problèmes se posent si GFS2 tente d'implémenter les verrous POSIX de NFS sur les nœuds d'une grappe. Pour les applications exécutées sur des clients NFS, les verrous POSIX localisés signifient que deux clients peuvent détenir le même verrou simultanément si les deux clients sont montés à partir de serveurs différents, ce qui pourrait entraîner une corruption des données. Si tous les clients montent NFS à partir d'un seul serveur, le problème de l'octroi indépendant des mêmes verrous par des serveurs distincts disparaît. Si vous n'êtes pas sûr de pouvoir monter votre système de fichiers avec l'option **localflocks**, vous ne devez pas utiliser cette option. Contactez immédiatement l'assistance de Red Hat pour discuter de la configuration appropriée afin d'éviter toute perte de données. L'exportation de GFS2 via NFS, bien que techniquement supportée dans certaines circonstances, n'est pas recommandée.

Pour toutes les autres applications GFS2 (non NFS), ne montez pas votre système de fichiers à l'aide de **localflocks**, afin que GFS2 gère les verrous et les blocs POSIX entre tous les nœuds de la grappe (à l'échelle de la grappe). Si vous spécifiez **localflocks** et que vous n'utilisez pas NFS, les autres nœuds de la grappe n'auront pas connaissance des verrous et des flocs POSIX des autres nœuds, ce qui les rendra peu sûrs dans un environnement en grappe

Outre les considérations relatives au verrouillage, vous devez tenir compte des éléments suivants lors de la configuration d'un service NFS sur un système de fichiers GFS2.

- Red Hat ne prend en charge que les configurations Red Hat High Availability Add-On utilisant NFSv3 avec verrouillage dans une configuration active/passive avec les caractéristiques suivantes. Cette configuration fournit une haute disponibilité (HA) pour le système de fichiers et réduit les temps d'arrêt du système puisqu'un nœud défaillant ne nécessite pas l'exécution de la commande **fsck** lors de la défaillance du serveur NFS d'un nœud à l'autre.
    - Le système de fichiers dorsal est un système de fichiers GFS2 fonctionnant sur une grappe de 2 à 16 nœuds.

- Un serveur NFSv3 est défini comme un service exportant l'intégralité du système de fichiers GFS2 à partir d'un seul nœud de cluster à la fois.
- Le serveur NFS peut basculer d'un nœud de cluster à un autre (configuration active/passive).
- Aucun accès au système de fichiers GFS2 n'est autorisé *except* via le serveur NFS. Cela concerne aussi bien l'accès local au système de fichiers GFS2 que l'accès via Samba ou Samba en grappe. L'accès local au système de fichiers via le nœud de cluster à partir duquel il est monté peut entraîner une corruption des données.
- Le système ne prend pas en charge les quotas NFS.

- L'option **fsid=** NFS est obligatoire pour les exportations NFS de GFS2.
- Si des problèmes surviennent avec votre cluster (par exemple, le cluster devient inquorate et la clôture n'est pas réussie), les volumes logiques clusterisés et le système de fichiers GFS2 seront gelés et aucun accès ne sera possible jusqu'à ce que le cluster soit quorate. Vous devez tenir compte de cette possibilité lorsque vous déterminez si une solution de basculement simple telle que celle définie dans cette procédure est la plus appropriée pour votre système.

2.5. SERVICE DE FICHIERS SAMBA (SMB OU WINDOWS) SUR GFS2

Vous pouvez utiliser le service de fichiers Samba (SMB ou Windows) à partir d'un système de fichiers GFS2 avec CTDB, qui permet des configurations actives/actives.

L'accès simultané aux données du partage Samba depuis l'extérieur de Samba n'est pas pris en charge. Il n'y a actuellement aucune prise en charge des baux de grappes GFS2, qui ralentissent le service de fichiers Samba. Pour plus d'informations sur les politiques de prise en charge de Samba, voir Politiques de prise en charge du stockage résilient RHEL - Politiques générales de ctdb et Politiques de prise en charge du stockage résilient RHEL - Exportation de contenus gfs2 via d'autres protocoles.

2.6. CONFIGURATION DES MACHINES VIRTUELLES POUR GFS2

Lors de l'utilisation d'un système de fichiers GFS2 avec une machine virtuelle, il est important que les paramètres de stockage de la VM sur chaque nœud soient configurés correctement afin de forcer la désactivation du cache. Par exemple, l'inclusion de ces paramètres pour **cache** et **io** dans le domaine **libvirt** devrait permettre à GFS2 de se comporter comme prévu.

```
<driver name='qemu' type='raw' cache='none' io='native'/>
```

Vous pouvez également configurer l'attribut **shareable** dans l'élément device. Cet attribut indique que l'appareil est censé être partagé entre les domaines (tant que l'hyperviseur et le système d'exploitation le prennent en charge). Si **shareable** est utilisé, **cache='no'** doit être utilisé pour ce dispositif.

2.7. ATTRIBUTION DE BLOCS

Même si les applications qui ne font qu'écrire des données ne se soucient généralement pas de savoir comment ou où un bloc est alloué, une certaine connaissance du fonctionnement de l'allocation des blocs peut vous aider à optimiser les performances.

2.7.1. Laisser de l'espace libre dans le système de fichiers

Lorsqu'un système de fichiers GFS2 est presque plein, l'allocateur de blocs commence à avoir du mal à

trouver de l'espace pour les nouveaux blocs à allouer. Par conséquent, les blocs alloués par l'allocateur ont tendance à se retrouver à la fin d'un groupe de ressources ou dans de minuscules tranches où la fragmentation des fichiers est beaucoup plus probable. Cette fragmentation des fichiers peut entraîner des problèmes de performance. En outre, lorsqu'un système de fichiers GFS2 est presque plein, l'allocateur de blocs GFS2 passe plus de temps à rechercher dans plusieurs groupes de ressources, ce qui ajoute une contention de verrouillage qui n'existerait pas nécessairement sur un système de fichiers disposant de beaucoup d'espace libre. Cela peut également entraîner des problèmes de performance.

Pour ces raisons, il est recommandé de ne pas utiliser un système de fichiers rempli à plus de 85 %, bien que ce chiffre puisse varier en fonction de la charge de travail.

2.7.2. Si possible, chaque nœud alloue ses propres fichiers

Lorsque vous développez des applications destinées à être utilisées avec les systèmes de fichiers GFS2, il est recommandé que chaque nœud alloue ses propres fichiers, si possible. En raison du fonctionnement du gestionnaire de verrous distribués (DLM), il y aura plus de conflits de verrous si tous les fichiers sont alloués par un nœud et que d'autres nœuds ont besoin d'ajouter des blocs à ces fichiers.

Le terme "maître des verrous" a été utilisé historiquement pour désigner un nœud qui est actuellement le coordinateur des demandes de verrous, qui proviennent d'un nœud local ou d'un nœud distant dans le cluster. Ce terme de coordinateur des demandes de verrouillage est légèrement trompeur, car il s'agit en réalité d'une ressource (dans la terminologie DLM) par rapport à laquelle les demandes de verrouillage sont soit mises en file d'attente, soit accordées, soit refusées. Dans le sens où ce terme est utilisé dans le DLM, il doit être considéré comme faisant référence au "premier parmi les égaux", puisque le DLM est un système pair-à-pair.

Dans la mise en œuvre du DLM du noyau Linux, le nœud sur lequel le verrou est utilisé pour la première fois devient le coordinateur des demandes de verrou, et ne change plus ensuite. Il s'agit d'un détail d'implémentation du DLM du noyau Linux et non d'une propriété des DLM en général. Il est possible qu'une future mise à jour permette à la coordination des demandes de verrouillage pour un verrou particulier de passer d'un nœud à l'autre.

L'endroit où les demandes de verrouillage sont coordonnées est transparent pour l'initiateur de la demande de verrouillage, sauf en ce qui concerne l'effet sur la latence de la demande. L'une des conséquences de la mise en œuvre actuelle est qu'en cas de déséquilibre de la charge de travail initiale (par exemple, un nœud parcourt l'ensemble du système de fichiers avant que d'autres n'exécutent des commandes d'E/S), il peut en résulter des temps de latence de verrouillage plus élevés pour les autres nœuds de la grappe que pour le nœud qui a effectué le premier balayage du système de fichiers.

Comme dans de nombreux systèmes de fichiers, l'allocateur GFS2 tente de maintenir les blocs d'un même fichier à proximité les uns des autres afin de réduire le déplacement des têtes de disque et d'améliorer les performances. Un nœud qui alloue des blocs à un fichier devra probablement utiliser et verrouiller les mêmes groupes de ressources pour les nouveaux blocs (à moins que tous les blocs de ce groupe de ressources ne soient utilisés). Le système de fichiers fonctionnera plus rapidement si le coordinateur des demandes de verrouillage pour le groupe de ressources contenant le fichier alloue ses blocs de données (il est plus rapide de demander au nœud qui a ouvert le fichier en premier d'écrire tous les nouveaux blocs).

2.7.3. Préallocation, si possible

Si les fichiers sont préalloués, l'allocation de blocs peut être évitée et le système de fichiers peut fonctionner plus efficacement. GFS2 inclut l'appel système **fallocate(1)**, que vous pouvez utiliser pour préallouer des blocs de données.

CHAPITRE 3. ADMINISTRATION DES SYSTÈMES DE FICHIERS GFS2

Il existe toute une série de commandes et d'options permettant de créer, de monter, de développer et de gérer les systèmes de fichiers GFS2.

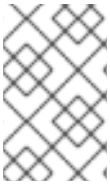
3.1. CRÉATION D'UN SYSTÈME DE FICHIERS GFS2

Vous créez un système de fichiers GFS2 à l'aide de la commande **mkfs.gfs2**. Un système de fichiers est créé sur un volume LVM activé.

3.1.1. La commande GFS2 mkfs

Les informations suivantes sont nécessaires pour exécuter la commande **mkfs.gfs2** afin de créer un système de fichiers GFS2 en cluster :

- Nom du protocole/module de verrouillage, qui est **lock_dlm** pour un cluster
- Nom du groupe
- Nombre de journaux (un journal est requis pour chaque nœud susceptible de monter le système de fichiers)



NOTE

Une fois que vous avez créé un système de fichiers GFS2 avec la commande **mkfs.gfs2**, vous ne pouvez pas réduire la taille du système de fichiers. Vous pouvez toutefois augmenter la taille d'un système de fichiers existant à l'aide de la commande **gfs2_grow**.

Le format de création d'un système de fichiers GFS2 en grappe est le suivant. Notez que Red Hat ne prend pas en charge l'utilisation de GFS2 en tant que système de fichiers à nœud unique.

```
mkfs.gfs2 -p lock_dlm -t ClusterName:FSName -j NumberJournals BlockDevice
```

Si vous préférez, vous pouvez créer un système de fichiers GFS2 en utilisant la commande **mkfs** avec le paramètre **-t** spécifiant un système de fichiers de type **gfs2**, suivi des options du système de fichiers GFS2.

```
mkfs -t gfs2 -p lock_dlm -t ClusterName:FSName -j NumberJournals BlockDevice
```

AVERTISSEMENT

Une mauvaise spécification du paramètre *ClusterName:FSName* peut entraîner une corruption du système de fichiers ou de l'espace de verrouillage.

ClusterName

Le nom du cluster pour lequel le système de fichiers GFS2 est créé.

FSName

Le nom du système de fichiers, qui peut comporter de 1 à 16 caractères. Ce nom doit être unique pour tous les systèmes de fichiers **lock_dlm** du cluster.

NumberJournals

Spécifie le nombre de journaux à créer par la commande **mkfs.gfs2**. Un journal est nécessaire pour chaque nœud qui monte le système de fichiers. Pour les systèmes de fichiers GFS2, des journaux supplémentaires peuvent être ajoutés ultérieurement sans augmenter la taille du système de fichiers.

BlockDevice

Spécifie un périphérique logique ou un autre périphérique de bloc

Le tableau suivant décrit les options de la commande **mkfs.gfs2** (drapeaux et paramètres).

Tableau 3.1. Options de commande :mkfs.gfs2

Drapeau	Paramètres	Description
-c	**Megabytes**	Fixe la taille initiale du fichier de modification des quotas de chaque journal à **Megabytes**.
-D		Active la sortie de débogage.
-h		Aide. Affiche les options disponibles.
-J	**Megabytes**	Spécifie la taille du journal en mégaoctets. La taille par défaut du journal est de 128 mégaoctets. La taille minimale est de 8 mégaoctets. Les journaux de plus grande taille améliorent les performances, bien qu'ils utilisent plus de mémoire que les journaux de plus petite taille.
-j	**Number**	Spécifie le nombre de journaux à créer par la commande **mkfs.gfs2**. Un journal est nécessaire pour chaque nœud qui monte le système de fichiers. Si cette option n'est pas spécifiée, un journal sera créé. Pour les systèmes de fichiers GFS2, vous pouvez ajouter des journaux supplémentaires ultérieurement sans augmenter la taille du système de fichiers.
-O		Empêche la commande **mkfs.gfs2** de demander une confirmation avant d'écrire le système de fichiers.

Drapeau	Paramètres	Description
-p	**LockProtoName**	* Indique le nom du protocole de fermeture à utiliser. Les protocoles de fermeture reconnus sont les suivants * **lock_dlm** - Le module de verrouillage standard, nécessaire pour un système de fichiers en cluster. * **lock_nolock** - Utilisé lorsque GFS2 agit comme un système de fichiers local (un seul nœud).
-q		Silence. Ne rien afficher.
-r	**Megabytes**	Spécifie la taille des groupes de ressources en mégaoctets. La taille minimale des groupes de ressources est de 32 mégaoctets. La taille maximale des groupes de ressources est de 2048 mégaoctets. Une taille de groupe de ressources importante peut améliorer les performances sur les systèmes de fichiers très volumineux. Si cette valeur n'est pas spécifiée, **mkfs.gfs2** choisit la taille du groupe de ressources en fonction de la taille du système de fichiers : les systèmes de fichiers de taille moyenne auront des groupes de ressources de 256 mégaoctets, et les systèmes de fichiers plus importants auront des groupes de ressources plus importants pour de meilleures performances.

Drapeau	Paramètres	Description
-t	**LockTableName**	* Un identifiant unique qui spécifie le champ de la table de verrouillage lorsque vous utilisez le protocole **lock_dlm**; le protocole **lock_nolock** n'utilise pas ce paramètre. * Ce paramètre est composé de deux parties séparées par deux points (sans espace) comme suit : **ClusterName:FSName**. * **ClusterName** est le nom de la grappe pour laquelle le système de fichiers GFS2 est créé ; seuls les membres de cette grappe sont autorisés à utiliser ce système de fichiers. * **FSName** le nom du système de fichiers peut comporter de 1 à 16 caractères et doit être unique parmi tous les systèmes de fichiers du cluster.
-V		Affiche des informations sur la version de la commande.

3.1.2. Création d'un système de fichiers GFS2

L'exemple suivant crée deux systèmes de fichiers GFS2. Pour ces deux systèmes de fichiers, lock_dlm` est le protocole de verrouillage que le système de fichiers utilise, puisqu'il s'agit d'un système de fichiers en grappe. Les deux systèmes de fichiers peuvent être utilisés dans le cluster nommé **alpha**.

Pour le premier système de fichiers, le nom du système de fichiers est **mydata1**. Il contient huit journaux et est créé sur **/dev/vg01/lvol0**. Pour le deuxième système de fichiers, le nom du système de fichiers est **mydata2**. Il contient huit journaux et a été créé sur **/dev/vg01/lvol1**.

```
# mkfs.gfs2 -p lock_dlm -t alpha:mydata1 -j 8 /dev/vg01/lvol0
# mkfs.gfs2 -p lock_dlm -t alpha:mydata2 -j 8 /dev/vg01/lvol1
```

3.2. MONTAGE D'UN SYSTÈME DE FICHIERS GFS2

Avant de pouvoir monter un système de fichiers GFS2, le système de fichiers doit exister, le volume où se trouve le système de fichiers doit être activé et les systèmes de clustering et de verrouillage doivent être démarrés. Une fois ces conditions remplies, vous pouvez monter le système de fichiers GFS2 comme n'importe quel système de fichiers Linux.

NOTE

Dans un environnement de production, vous devez toujours utiliser Pacemaker pour gérer le système de fichiers GFS2 plutôt que de monter manuellement le système de fichiers à l'aide d'une commande **mount**, car cela peut entraîner des problèmes lors de l'arrêt du système.

Pour manipuler les ACL de fichiers, vous devez monter le système de fichiers avec l'option de montage **-o acl**. Si un système de fichiers est monté sans l'option de montage **-o acl**, les utilisateurs sont autorisés à visualiser les listes de contrôle d'accès (avec **getfacl**), mais ne sont pas autorisés à les définir (avec **setfacl**).

3.2.1. Montage d'un système de fichiers GFS2 sans options spécifiées

Dans cet exemple, le système de fichiers GFS2 sur **/dev/vg01/lvol0** est monté sur le répertoire **/mygfs2**.

```
# mount /dev/vg01/lvol0 /mygfs2
```

3.2.2. Montage d'un système de fichiers GFS2 spécifiant des options de montage

Voici le format de la commande de montage d'un système de fichiers GFS2 qui spécifie les options de montage.

```
mount BlockDevice MountPoint -o option
```

BlockDevice

Spécifie le périphérique de bloc où réside le système de fichiers GFS2.

MountPoint

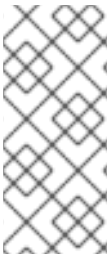
Spécifie le répertoire dans lequel le système de fichiers GFS2 doit être monté.

L'argument **-o option** se compose d'options spécifiques à GFS2 ou d'options Linux standard acceptables **mount -o**, ou d'une combinaison des deux. Les paramètres **option** multiples sont séparés par une virgule et sans espace.

NOTE

La commande **mount** est une commande système Linux. Outre ces options spécifiques à GFS2, vous pouvez utiliser d'autres options standard de la commande **mount** (par exemple, **-r**). Pour plus d'informations sur les autres options de la commande Linux **mount**, consultez la page de manuel Linux **mount**.

Le tableau suivant décrit les valeurs **-o option** spécifiques à GFS2 qui peuvent être transmises à GFS2 au moment du montage.



NOTE

Ce tableau inclut des descriptions d'options qui sont utilisées avec des systèmes de fichiers locaux uniquement. Notez cependant que Red Hat ne prend pas en charge l'utilisation de GFS2 en tant que système de fichiers à nœud unique. Red Hat continuera à prendre en charge les systèmes de fichiers GFS2 à nœud unique pour le montage d'instantanés de systèmes de fichiers en grappe (par exemple, à des fins de sauvegarde).

Tableau 3.2. Options de montage spécifiques au GFS2

Option	Description
acl	Permet de manipuler les ACL des fichiers. Si un système de fichiers est monté sans l'option **acl** mount, les utilisateurs sont autorisés à consulter les listes de contrôle d'accès (avec **getfacl**), mais ne sont pas autorisés à les définir (avec **setfacl**).
data=[ordered|writeback]	Lorsque **data=ordered** est défini, les données utilisateur modifiées par une transaction sont transférées sur le disque avant que la transaction ne soit validée sur le disque. Cela devrait empêcher l'utilisateur de voir des blocs non initialisés dans un fichier après un crash. Lorsque le mode **data=writeback** est activé, les données utilisateur sont écrites sur le disque à tout moment après qu'elles ont été supprimées. Ce mode n'offre pas la même garantie de cohérence que le mode **ordered**, mais il devrait être légèrement plus rapide pour certaines charges de travail. La valeur par défaut est le mode **ordered**.
ignore_local_fs **Caution:** Cette option ne doit pas être utilisée lorsque les systèmes de fichiers GFS2 sont partagés.	Force GFS2 à traiter le système de fichiers comme un système de fichiers multi-hôtes. Par défaut, l'utilisation de **lock_nolock** active automatiquement l'indicateur **localflocks**.
localflocks **Caution:** Cette option ne doit pas être utilisée lorsque les systèmes de fichiers GFS2 sont partagés.	Indique à GFS2 de laisser la couche VFS (système de fichiers virtuels) s'occuper de tous les flocages et fcntl. Le drapeau **localflocks** est automatiquement activé par **lock_nolock**.
lockproto=LockModuleName	Permet à l'utilisateur de spécifier le protocole de verrouillage à utiliser avec le système de fichiers. Si **LockModuleName** n'est pas spécifié, le nom du protocole de verrouillage est lu dans le superbloc du système de fichiers.
locktable=LockTableName	Permet à l'utilisateur de spécifier la table de verrouillage à utiliser avec le système de fichiers.
quota=[off/account/on]	Active ou désactive les quotas pour un système de fichiers. Si les quotas sont définis à l'état **account**, les statistiques d'utilisation par UID/GID sont correctement maintenues par le système de fichiers ; les valeurs de limite et d'avertissement sont ignorées. La valeur par défaut est **off**.

Option	Description
errors=panic|withdraw	Lorsque **errors=panic** est spécifié, les erreurs du système de fichiers provoquent une panique du noyau. Lorsque **errors=withdraw** est spécifié, ce qui est le comportement par défaut, les erreurs du système de fichiers entraînent le retrait du système de fichiers et le rendent inaccessible jusqu'au prochain redémarrage ; dans certains cas, le système peut continuer à fonctionner.
discard/nodiscard	GFS2 génère des demandes d'E/S "discard" pour les blocs qui ont été libérés. Celles-ci peuvent être utilisées par le matériel adéquat pour mettre en œuvre le "thin provisioning" et d'autres schémas similaires.
barrier/nobarrier	Permet à GFS2 d'envoyer des barrières d'E/S lors de la vidange du journal. La valeur par défaut est **on**. Cette option est automatiquement désactivée à l'adresse **off** si le périphérique sous-jacent ne prend pas en charge les barrières d'entrée/sortie. L'utilisation de barrières d'E/S avec GFS2 est fortement recommandée à tout moment, sauf si le périphérique bloc est conçu de manière à ne pas perdre le contenu de son cache d'écriture (par exemple, s'il se trouve sur un onduleur ou s'il n'a pas de cache d'écriture).
quota_quantum=*secs*	Définit le nombre de secondes pendant lesquelles une modification des informations relatives aux quotas peut rester en attente sur un nœud avant d'être écrite dans le fichier de quotas. Il s'agit de la méthode préférée pour définir ce paramètre. La valeur est un nombre entier de secondes supérieur à zéro. La valeur par défaut est de 60 secondes. Des paramètres plus courts entraînent des mises à jour plus rapides des informations relatives au quota de paresseux et réduisent la probabilité que quelqu'un dépasse son quota. Des paramètres plus longs rendent les opérations du système de fichiers impliquant des quotas plus rapides et plus efficaces.
statfs_quantum=*secs*	La valeur 0 pour **statfs_quantum** est la meilleure façon de définir la version lente de **statfs**. La valeur par défaut est de 30 secondes, ce qui fixe le délai maximum avant que les modifications apportées à **statfs** ne soient synchronisées avec le fichier maître **statfs**. Cette valeur peut être ajustée pour permettre des valeurs **statfs** plus rapides et moins précises ou des valeurs plus lentes et plus précises. Lorsque cette option est fixée à 0, **statfs** indique toujours les valeurs réelles.

Option	Description
statfs_percent=*value*	Limite le pourcentage maximum de changement dans les informations **statfs** au niveau local avant qu'elles ne soient synchronisées avec le fichier principal **statfs**, même si le délai n'a pas expiré. Si la valeur de **statfs_quantum** est 0, ce paramètre est ignoré.

3.2.3. Démontage d'un système de fichiers GFS2

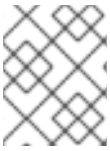
Les systèmes de fichiers GFS2 qui ont été montés manuellement plutôt qu'automatiquement par Pacemaker ne seront pas connus du système lorsque les systèmes de fichiers sont démontés à l'arrêt du système. Par conséquent, l'agent de ressources GFS2 ne démontera pas le système de fichiers GFS2. Après l'arrêt de l'agent de ressources GFS2, le processus d'arrêt standard arrête tous les processus utilisateur restants, y compris l'infrastructure du cluster, et tente de démonter le système de fichiers. Ce démontage échouera sans l'infrastructure de cluster et le système se bloquera.

Pour éviter que le système ne se bloque lorsque les systèmes de fichiers GFS2 sont démontés, vous devez effectuer l'une des opérations suivantes :

- Utilisez toujours Pacemaker pour gérer le système de fichiers GFS2.
- Si un système de fichiers GFS2 a été monté manuellement à l'aide de la commande **mount**, veillez à démonter le système de fichiers manuellement à l'aide de la commande **umount** avant de redémarrer ou d'arrêter le système.

Si votre système de fichiers se bloque pendant qu'il est démonté lors de l'arrêt du système dans ces circonstances, procédez à un redémarrage du matériel. Il est peu probable que des données soient perdues puisque le système de fichiers est synchronisé plus tôt dans le processus d'arrêt.

Le système de fichiers GFS2 peut être démonté de la même manière que n'importe quel système de fichiers Linux, à l'aide de la commande **umount**.



NOTE

La commande **umount** est une commande système de Linux. Des informations sur cette commande sont disponibles dans les pages de manuel de la commande Linux **umount**.

Utilisation

```
montage MountPoint
```

MountPoint

Spécifie le répertoire dans lequel le système de fichiers GFS2 est actuellement monté.

3.3. SAUVEGARDE D'UN SYSTÈME DE FICHIERS GFS2

Il est important d'effectuer des sauvegardes régulières de votre système de fichiers GFS2 en cas d'urgence, quelle que soit la taille de votre système de fichiers. De nombreux administrateurs système se sentent en sécurité parce qu'ils sont protégés par des systèmes RAID, multipath, mirroring, snapshots et autres formes de redondance, mais il n'y a pas de sécurité suffisante.

La création d'une sauvegarde peut s'avérer problématique car le processus de sauvegarde d'un nœud ou d'un ensemble de nœuds implique généralement la lecture séquentielle de l'ensemble du système de fichiers. Si cette opération est effectuée à partir d'un seul nœud, celui-ci conservera toutes les informations en mémoire cache jusqu'à ce que les autres nœuds de la grappe commencent à demander des verrous. L'exécution de ce type de programme de sauvegarde pendant que la grappe fonctionne aura un impact négatif sur les performances.

L'abandon des caches une fois la sauvegarde terminée réduit le temps nécessaire aux autres nœuds pour reprendre possession de leurs verrous/caches de cluster. Cette solution n'est toutefois pas idéale, car les autres nœuds auront cessé de mettre en cache les données qu'ils mettaient en cache avant le début du processus de sauvegarde. Vous pouvez supprimer les caches à l'aide de la commande suivante une fois la sauvegarde terminée :

```
echo -n 3 > /proc/sys/vm/drop_caches
```

Il est plus rapide que chaque nœud de la grappe sauvegarde ses propres fichiers afin que la tâche soit répartie entre les nœuds. Vous pouvez y parvenir avec un script qui utilise la commande **rsync** sur des répertoires spécifiques à chaque nœud.

Red Hat recommande d'effectuer une sauvegarde GFS2 en créant un instantané matériel sur le SAN, en présentant l'instantané à un autre système et en le sauvegardant à cet endroit. Le système de sauvegarde doit monter l'instantané avec **-o lockproto=lock_nolock** puisqu'il ne sera pas dans un cluster.

3.4. SUSPENDRE L'ACTIVITÉ D'UN SYSTÈME DE FICHIERS GFS2

Vous pouvez suspendre l'activité d'écriture sur un système de fichiers à l'aide de la commande **dmsetup suspend**. La suspension de l'activité d'écriture permet d'utiliser des instantanés de périphériques matériels pour capturer le système de fichiers dans un état cohérent. La commande **dmsetup resume** met fin à la suspension.

Le format de la commande de suspension de l'activité d'un système de fichiers GFS2 est le suivant.

```
dmsetup suspend MountPoint
```

Cet exemple suspend les écritures sur le système de fichiers /**mygfs2**.

```
# dmsetup suspend /mygfs2
```

Le format de la commande de fin de suspension d'activité sur un système de fichiers GFS2 est le suivant.

```
dmsetup resume MountPoint
```

Cet exemple met fin à la suspension des écritures sur le système de fichiers /**mygfs2**.

```
# dmsetup resume /mygfs2
```

3.5. DÉVELOPPEMENT D'UN SYSTÈME DE FICHIERS GFS2

La commande **gfs2_grow** est utilisée pour étendre un système de fichiers GFS2 après que le périphérique où réside le système de fichiers a été étendu. L'exécution de la commande **gfs2_grow** sur un système de fichiers GFS2 existant remplit tout l'espace libre entre l'extrémité actuelle du système de

fichiers et l'extrémité du périphérique avec une extension de système de fichiers GFS2 nouvellement initialisée. Tous les nœuds du cluster peuvent alors utiliser l'espace de stockage supplémentaire qui a été ajouté.

NOTE

Vous ne pouvez pas réduire la taille d'un système de fichiers GFS2.

La commande **gfs2_grow** doit être exécutée sur un système de fichiers monté. La procédure suivante permet d'augmenter la taille du système de fichiers GFS2 dans un cluster qui est monté sur le volume logique **shared_vg/shared_lv1** avec un point de montage de **/mnt/gfs2**.

Procédure

1. Effectuer une sauvegarde des données du système de fichiers.
2. Si vous ne connaissez pas le volume logique utilisé par le système de fichiers à développer, vous pouvez le déterminer en exécutant la commande **df *mountpoint*** pour le déterminer. Celle-ci affichera le nom du périphérique dans le format suivant :
   /dev/mapper/*vg-lv*

   Par exemple, le nom de l'appareil **/dev/mapper/shared_vg-shared_lv1** indique que le volume logique est **shared_vg/shared_lv1**.
3. Sur un nœud de la grappe, développez le volume sous-jacent de la grappe à l'aide de la commande **lvextend**.

```
# lvextend -L+1G shared_vg/shared_lv1
Size of logical volume shared_vg/shared_lv1 changed from 5.00 GiB (1280 extents) to 6.00 GiB (1536 extents).
WARNING: extending LV with a shared lock, other hosts may require LV refresh.
Logical volume shared_vg/shared_lv1 successfully resized.
```

4. Sur un nœud de la grappe, augmentez la taille du système de fichiers GFS2. Ne pas étendre le système de fichiers si le volume logique n'a pas été rafraîchi sur tous les nœuds, sinon les données du système de fichiers risquent de devenir indisponibles dans l'ensemble du cluster.

```
# gfs2_grow /mnt/gfs2
FS: Mount point:             /mnt/gfs2
FS: Device:                  /dev/mapper/shared_vg-shared_lv1
FS: Size:                    1310719 (0x13ffff)
DEV: Length:                 1572864 (0x180000)
The file system will grow by 1024MB.
gfs2_grow complete.
```

5. Exécutez la commande **df** sur tous les nœuds pour vérifier que le nouvel espace est désormais disponible dans le système de fichiers. Notez qu'il peut s'écouler jusqu'à 30 secondes avant que la commande **df** n'affiche la même taille de système de fichiers sur tous les nœuds

```
# df -h /mnt/gfs2]
Filesystem                      Size  Used Avail Use% Mounted on
/dev/mapper/shared_vg-shared_lv1  6.0G  4.5G  1.6G  75% /mnt/gfs2
```

3.6. AJOUT DE JOURNAUX À UN SYSTÈME DE FICHIERS GFS2

GFS2 nécessite un journal pour chaque nœud d'une grappe qui doit monter le système de fichiers. Si vous ajoutez des nœuds supplémentaires à la grappe, vous pouvez ajouter des journaux à un système de fichiers GFS2 à l'aide de la commande **gfs2_jadd**. Vous pouvez ajouter des journaux à un système de fichiers GFS2 de manière dynamique à tout moment sans étendre le volume logique sous-jacent. La commande **gfs2_jadd** doit être exécutée sur un système de fichiers monté, mais sur un seul nœud du cluster. Tous les autres nœuds détectent que l'extension a eu lieu.

NOTE

Si un système de fichiers GFS2 est plein, la commande **gfs2_jadd** échouera, même si le volume logique contenant le système de fichiers a été étendu et est plus grand que le système de fichiers. En effet, dans un système de fichiers GFS2, les journaux sont des fichiers simples et non des métadonnées intégrées, de sorte que la simple extension du volume logique sous-jacent n'offrira pas d'espace pour les journaux.

Avant d'ajouter des journaux à un système de fichiers GFS2, vous pouvez savoir combien de journaux le système de fichiers GFS2 contient actuellement à l'aide de la commande **gfs2_edit -p jindex**, comme dans l'exemple suivant :

```
# gfs2_edit -p jindex /dev/sasdrives/scratch|grep journal
  3/3 [fc7745eb] 4/25 (0x4/0x19): File    journal0
  4/4 [8b70757d] 5/32859 (0x5/0x805b): File    journal1
  5/5 [127924c7] 6/65701 (0x6/0x100a5): File    journal2
```

Le format de la commande de base pour ajouter des journaux à un système de fichiers GFS2 est le suivant.

```
gfs2_jadd -j Number MountPoint
```

Number

Spécifie le nombre de nouveaux journaux à ajouter.

MountPoint

Spécifie le répertoire dans lequel le système de fichiers GFS2 est monté.

Dans cet exemple, un journal est ajouté au système de fichiers du répertoire **/mygfs2**.

```
# gfs2_jadd -j 1 /mygfs2
```

CHAPITRE 4. GESTION DES QUOTAS GFS2

Les quotas de système de fichiers sont utilisés pour limiter la quantité d'espace de système de fichiers qu'un utilisateur ou un groupe peut utiliser. Un utilisateur ou un groupe n'a pas de limite de quota tant qu'il n'en a pas défini une. Lorsqu'un système de fichiers GFS2 est monté avec l'option **quota=on** ou **quota=account**, GFS2 garde une trace de l'espace utilisé par chaque utilisateur et chaque groupe, même si aucune limite n'a été fixée. GFS2 met à jour les informations relatives aux quotas de manière transactionnelle, de sorte que les pannes de système n'exigent pas la reconstitution de l'utilisation des quotas.

Pour éviter un ralentissement des performances, un nœud GFS2 ne synchronise les mises à jour du fichier de quotas que périodiquement. La comptabilisation floue des quotas peut permettre aux utilisateurs ou aux groupes de dépasser légèrement la limite fixée. Pour minimiser ce phénomène, GFS2 réduit dynamiquement la période de synchronisation à l'approche d'une limite de quota stricte.

NOTE

GFS2 prend en charge les fonctions de quota standard de Linux. Pour les utiliser, vous devez installer le fichier **quota** RPM. C'est la méthode préférée pour administrer les quotas sur GFS2 et elle doit être utilisée pour tous les nouveaux déploiements de GFS2 utilisant des quotas.

Pour plus d'informations sur les quotas de disque, consultez les pages **man** des commandes suivantes :

- **quotacheck**
- **edquota**
- **repquota**
- **quota**

4.1. CONFIGURATION DES QUOTAS DE DISQUE GFS2

La mise en œuvre des quotas de disque pour les systèmes de fichiers GFS2 s'effectue en trois étapes.

Les étapes à suivre pour mettre en œuvre les quotas de disque sont les suivantes :

1. Configurer les quotas en mode exécution ou en mode comptabilité.
2. Initialiser le fichier de la base de données des quotas avec les informations sur l'utilisation actuelle des blocs.
3. Attribuer des politiques de quotas. (En mode comptabilité, ces politiques ne sont pas appliquées)

Chacune de ces étapes est examinée en détail dans les sections suivantes.

4.1.1. Mise en place de quotas en mode exécution ou comptabilité

Dans les systèmes de fichiers GFS2, les quotas sont désactivés par défaut. Pour activer les quotas pour un système de fichiers, montez le système de fichiers en spécifiant l'option **quota=on**.

Pour monter un système de fichiers avec les quotas activés, spécifiez **quota=on** pour l'argument **options** lors de la création de la ressource du système de fichiers GFS2 dans un cluster. Par exemple, la

commande suivante indique que la ressource GFS2 **Filesystem** en cours de création sera montée avec les quotas activés.

```
# pcs resource create gfs2mount Filesystem options="quota=on" device=BLOCKDEVICE
directory=MOUNTPOINT fstype=gfs2 clone
```

Il est possible de suivre l'utilisation du disque et de gérer la comptabilité des quotas pour chaque utilisateur et groupe sans appliquer les valeurs de limite et d'avertissement. Pour ce faire, montez le système de fichiers en spécifiant l'option **quota=account**.

Pour monter un système de fichiers avec des quotas désactivés, spécifiez **quota=off** pour l'argument **options** lors de la création de la ressource de système de fichiers GFS2 dans un cluster.

4.1.2. Création des fichiers de la base de données des quotas

Une fois que chaque système de fichiers compatible avec les quotas est monté, le système est capable de fonctionner avec des quotas de disque. Toutefois, le système de fichiers lui-même n'est pas encore prêt à prendre en charge les quotas. L'étape suivante consiste à exécuter la commande **quotacheck**.

La commande **quotacheck** examine les systèmes de fichiers à quotas et construit un tableau de l'utilisation actuelle du disque par système de fichiers. Ce tableau est ensuite utilisé pour mettre à jour la copie de l'utilisation du disque du système d'exploitation. En outre, les fichiers de quotas de disque du système de fichiers sont mis à jour.

Pour créer les fichiers de quotas sur le système de fichiers, utilisez les options **-u** et **-g** de la commande **quotacheck**; ces deux options doivent être spécifiées pour que les quotas d'utilisateurs et de groupes soient initialisés. Par exemple, si les quotas sont activés pour le système de fichiers /**home**, créez les fichiers dans le répertoire /**home**:

```
# quotacheck -ug /home
```

4.1.3. Attribution de quotas par utilisateur

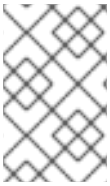
La dernière étape consiste à attribuer les quotas de disque à l'aide de la commande **edquota**. Notez que si vous avez monté votre système de fichiers en mode comptabilité (avec l'option **quota=account** spécifiée), les quotas ne sont pas appliqués.

Pour configurer le quota d'un utilisateur, exécutez la commande suivante en tant que root dans une invite de l'interpréteur de commandes :

```
# edquota username
```

Effectuez cette étape pour chaque utilisateur qui a besoin d'un quota. Par exemple, si un quota est activé pour la partition /**home** (/**dev**/**VolGroup00**/**LogVol02** dans l'exemple ci-dessous) et que la commande **edquota testuser** est exécutée, l'éditeur configuré par défaut pour le système affiche ce qui suit :

```
Disk quotas for user testuser (uid 501):
Filesystem                blocks   soft   hard   inodes   soft   hard
/dev/VolGroup00/LogVol02  440436      0      0
```



NOTE

L'éditeur de texte défini par la variable d'environnement **EDITOR** est utilisé par **edquota**. Pour changer d'éditeur, définissez la variable d'environnement **EDITOR** dans votre fichier ~/**.bash_profile** avec le chemin complet de l'éditeur de votre choix.

La première colonne est le nom du système de fichiers pour lequel un quota est activé. La deuxième colonne indique le nombre de blocs actuellement utilisés par l'utilisateur. Les deux colonnes suivantes sont utilisées pour définir des limites de blocs souples et dures pour l'utilisateur sur le système de fichiers.

La limite souple des blocs définit la quantité maximale d'espace disque qui peut être utilisée.

La limite des blocs durs est la quantité maximale absolue d'espace disque qu'un utilisateur ou un groupe peut utiliser. Une fois cette limite atteinte, aucun espace disque supplémentaire ne peut être utilisé.

Le système de fichiers GFS2 ne gère pas de quotas pour les inodes. Ces colonnes ne s'appliquent donc pas aux systèmes de fichiers GFS2 et seront vides.

Si l'une des valeurs est fixée à 0, cette limite n'est pas définie. Dans l'éditeur de texte, modifiez les limites. Par exemple :

```
Disk quotas for user testuser (uid 501):
Filesystem              blocks    soft    hard    inodes   soft   hard
/dev/VolGroup00/LogVol02  440436   500000   550000
```

Pour vérifier que le quota de l'utilisateur a été défini, utilisez la commande suivante :

```
# quota testuser
```

Vous pouvez également définir des quotas à partir de la ligne de commande avec la commande **setquota**. Pour plus d'informations sur la commande **setquota**, consultez la page de manuel **setquota**(8).

4.1.4. Attribution de quotas par groupe

Les quotas peuvent également être attribués par groupe. Notez que si vous avez monté votre système de fichiers en mode comptabilité (avec l'option **account=on** spécifiée), les quotas ne sont pas appliqués.

Pour définir un quota de groupe pour le groupe **devel** (le groupe doit exister avant de définir le quota de groupe), utilisez la commande suivante :

```
# edquota -g devel
```

Cette commande affiche le quota existant pour le groupe dans l'éditeur de texte :

```
Disk quotas for group devel (gid 505):
Filesystem              blocks   soft    hard    inodes   soft   hard
/dev/VolGroup00/LogVol02  440400       0       0
```

Le système de fichiers GFS2 ne gère pas de quotas pour les inodes. Ces colonnes ne s'appliquent donc pas aux systèmes de fichiers GFS2 et seront vides. Modifiez les limites, puis enregistrez le fichier.

Pour vérifier que le quota de groupe a été défini, utilisez la commande suivante :

```
$ quota -g devel
```

4.2. GESTION DES QUOTAS DE DISQUES GFS2

Si des quotas sont mis en place, ils nécessitent une certaine maintenance, principalement sous la forme d'une surveillance pour voir si les quotas sont dépassés et pour s'assurer que les quotas sont exacts.

Si les utilisateurs dépassent leurs quotas de façon répétée ou atteignent régulièrement leurs limites, l'administrateur système a plusieurs choix à faire en fonction du type d'utilisateurs et de l'impact de l'espace disque sur leur travail. L'administrateur peut soit aider l'utilisateur à déterminer comment utiliser moins d'espace disque, soit augmenter le quota de l'utilisateur.

Vous pouvez créer un rapport sur l'utilisation du disque en exécutant l'utilitaire **repquota**. Par exemple, la commande **repquota /home** produit le résultat suivant :

```
*** Report for user quotas on device /dev/mapper/VolGroup00-LogVol02
Block grace time: 7days; Inode grace time: 7days
  Block limits   File limits
User  used soft hard grace used soft hard grace
----------------------------------------------------------------------
root      --      36       0       0              4     0     0
kristin   --     540       0       0            125     0     0
testuser  --  440400  500000  550000          37418     0     0
```

Pour afficher le rapport d'utilisation du disque pour tous les systèmes de fichiers à quotas (option **-a**), utilisez la commande suivante :

```
# repquota -a
```

Le message **--** affiché après chaque utilisateur est un moyen rapide de déterminer si les limites de blocage ont été dépassées. Si la limite de blocs est dépassée, un apparaît à la place du premier **-** dans la sortie. Le deuxième **-** indique la limite d'inode, mais les systèmes de fichiers GFS2 ne prennent pas en charge les limites d'inode ; ce caractère restera donc **-**. Les systèmes de fichiers GFS2 ne prennent pas en charge le délai de grâce, de sorte que la colonne **grace** reste vide.

Notez que la commande **repquota** n'est pas prise en charge par NFS, quel que soit le système de fichiers sous-jacent.

4.3. MAINTENIR L'EXACTITUDE DES QUOTAS DE DISQUES GFS2 AVEC LA COMMANDE QUOTACHECK

Si vous activez les quotas sur votre système de fichiers après une période pendant laquelle les quotas étaient désactivés, vous devez exécuter la commande **quotacheck** pour créer, vérifier et réparer les fichiers de quotas. En outre, vous pouvez exécuter la commande **quotacheck** si vous pensez que vos fichiers de quotas ne sont pas exacts, comme cela peut se produire lorsqu'un système de fichiers n'est pas démonté proprement après une panne du système.

Pour plus d'informations sur la commande **quotacheck**, voir la page de manuel **quotacheck(8)**.

NOTE

Exécutez **quotacheck** lorsque le système de fichiers est relativement inactif sur tous les nœuds, car l'activité du disque peut affecter les valeurs de quota calculées.

4.4. SYNCHRONISATION DES QUOTAS AVEC LA COMMANDE QUOTASYNC

GFS2 stocke toutes les informations relatives aux quotas dans son propre fichier interne sur le disque. Un nœud GFS2 ne met pas à jour ce fichier de quotas à chaque écriture du système de fichiers ; par défaut, il met à jour le fichier de quotas une fois toutes les 60 secondes. Cela est nécessaire pour éviter les conflits entre les nœuds qui écrivent dans le fichier de quotas, ce qui entraînerait un ralentissement des performances.

Lorsqu'un utilisateur ou un groupe approche de sa limite de quota, GFS2 réduit dynamiquement l'intervalle de temps entre les mises à jour des fichiers de quota afin d'empêcher le dépassement de la limite. Le délai normal entre les synchronisations de quotas est un paramètre réglable, **quota_quantum**. Vous pouvez le modifier par rapport à sa valeur par défaut de 60 secondes à l'aide de l'option de montage **quota_quantum=**, comme décrit dans le tableau "Options de montage spécifiques à GFS2" dans Montage d'un système de fichiers GFS2 qui spécifie les options de montage .

Le paramètre **quota_quantum** doit être défini sur chaque nœud et à chaque fois que le système de fichiers est monté. Les modifications apportées au paramètre **quota_quantum** ne sont pas persistantes lors des démontages. Vous pouvez mettre à jour la valeur **quota_quantum** à l'aide de la commande **mount -o remount**.

Vous pouvez utiliser la commande **quotasync** pour synchroniser les informations de quotas d'un nœud avec le fichier de quotas sur disque entre les mises à jour automatiques effectuées par GFS2. Utilisation **Synchronizing Quota Information**

```
quotasync [-ug] -a|mountpoint...
```

u

Synchroniser les fichiers de quotas d'utilisateurs.

g

Synchroniser les fichiers de quotas de groupe

a

Synchronise tous les systèmes de fichiers qui sont actuellement compatibles avec les quotas et qui supportent la synchronisation. Lorsque -a est absent, un point de montage du système de fichiers doit être spécifié.

mountpoint

Spécifie le système de fichiers GFS2 auquel les actions s'appliquent.

Vous pouvez régler le délai entre les synchronisations en spécifiant une option de montage **quota-quantum**.

```
# mount -o quota_quantum=secs,remount BlockDevice MountPoint
```

MountPoint

Spécifie le système de fichiers GFS2 auquel les actions s'appliquent.

secs

Spécifie le nouveau délai entre les synchronisations régulières des fichiers de quotas par GFS2. Des valeurs plus petites peuvent augmenter les conflits et ralentir les performances.

L'exemple suivant synchronise tous les quotas sales mis en cache du nœud sur lequel il est exécuté avec le fichier de quotas sur disque pour le système de fichiers **/mnt/mygfs2**.

```
# quotasync -ug /mnt/mygfs2
```

L'exemple suivant modifie le délai par défaut entre les mises à jour régulières des fichiers de quotas à une heure (3600 secondes) pour le système de fichiers **/mnt/mygfs2** lors du remontage de ce système de fichiers sur le volume logique **/dev/volgroup/logical_volume**.

```
# mount -o quota_quantum=3600,remount /dev/volgroup/logical_volume /mnt/mygfs2
```

CHAPITRE 5. RÉPARATION DU SYSTÈME DE FICHIERS GFS2

Lorsque des nœuds tombent en panne alors que le système de fichiers est monté, la journalisation du système de fichiers permet une récupération rapide. Toutefois, si une unité de stockage perd de l'énergie ou est physiquement déconnectée, une corruption du système de fichiers peut se produire (la journalisation ne peut pas être utilisée pour récupérer les défaillances du sous-système de stockage). (La journalisation ne peut pas être utilisée pour récupérer les défaillances du sous-système de stockage.) Lorsque ce type de corruption se produit, vous pouvez récupérer le système de fichiers GFS2 à l'aide de la commande **fsck.gfs2**.

IMPORTANT

La commande **fsck.gfs2** ne doit être exécutée que sur un système de fichiers qui est démonté de tous les nœuds. Lorsque le système de fichiers est géré en tant que ressource de cluster Pacemaker, vous pouvez désactiver la ressource du système de fichiers, ce qui a pour effet de démonter le système de fichiers. Après avoir exécuté la commande **fsck.gfs2**, vous réactivez la ressource du système de fichiers. La valeur *timeout* spécifiée avec l'option **--wait** de la commande **pcs resource disable** indique une valeur en secondes.

```
pcs resource disable --wait=timeoutvalue resource_id
[fsck.gfs2]
pcs resource enable resource_id
```

Pour garantir que la commande **fsck.gfs2** ne s'exécute pas sur un système de fichiers GFS2 au moment du démarrage, vous pouvez définir le paramètre **run_fsck** de l'argument **options** lors de la création de la ressource du système de fichiers GFS2 dans un cluster. En spécifiant **"run_fsck=no"**, vous indiquez que vous ne devez pas exécuter la commande **fsck**.

5.1. DÉTERMINATION DE LA MÉMOIRE NÉCESSAIRE À L'EXÉCUTION DE FSCK.GFS2

L'exécution de la commande **fsck.gfs2** peut nécessiter de la mémoire système en plus de la mémoire utilisée pour le système d'exploitation et le noyau. Les systèmes de fichiers de grande taille, en particulier, peuvent nécessiter de la mémoire supplémentaire pour exécuter cette commande.

Le tableau suivant indique les valeurs approximatives de mémoire qui peuvent être nécessaires pour exécuter les systèmes de fichiers **fsck.gfs2** sur des systèmes de fichiers GFS2 d'une taille de 1 To, 10 To et 100 To avec une taille de bloc de 4K.

Taille du système de fichiers GFS2	Mémoire approximative nécessaire à l'exécution**fsck.gfs2**
1 TB	0.16 GO
10 TB	1.6 GB
100 TB	16 GB

Notez qu'une taille de bloc plus petite pour le système de fichiers nécessiterait une plus grande quantité de mémoire. Par exemple, les systèmes de fichiers GFS2 avec une taille de bloc de 1K nécessiteraient quatre fois la quantité de mémoire indiquée dans ce tableau.

5.2. RÉPARATION D'UN SYSTÈME DE FICHIERS GFS2

Le format de la commande **fsck.gfs2** pour réparer un système de fichiers GFS2 est le suivant :

```
fsck.gfs2 -y BlockDevice
```

-y

L'option **-y** permet de répondre à toutes les questions par **yes**. Si l'option **-y** est spécifiée, la commande **fsck.gfs2** ne vous demande pas de réponse avant d'effectuer des modifications.

BlockDevice

Spécifie le périphérique de bloc où réside le système de fichiers GFS2.

Dans cet exemple, le système de fichiers GFS2 résidant sur le périphérique de bloc **/dev/testvg/testlv** est réparé. Toutes les demandes de réparation reçoivent automatiquement la réponse suivante : **yes**.

```
# fsck.gfs2 -y /dev/testvg/testlv
Initializing fsck
Validating Resource Group index.
Level 1 RG check.
(level 1 passed)
Clearing journals (this may take a while)...
Journals cleared.
Starting pass1
Pass1 complete
Starting pass1b
Pass1b complete
Starting pass1c
Pass1c complete
Starting pass2
Pass2 complete
Starting pass3
Pass3 complete
Starting pass4
Pass4 complete
Starting pass5
Pass5 complete
Writing changes to disk
fsck.gfs2 complete
```

CHAPITRE 6. AMÉLIORER LES PERFORMANCES DE GFS2

Il existe de nombreux aspects de la configuration de GFS2 que vous pouvez analyser pour améliorer les performances du système de fichiers.

Pour des recommandations générales concernant le déploiement et la mise à niveau de clusters Red Hat Enterprise Linux à l'aide du module complémentaire de haute disponibilité et de Red Hat Global File System 2 (GFS2), consultez l'article Red Hat Enterprise Linux Cluster, High Availability, and GFS Deployment Best Practices sur le portail client Red Hat.

6.1. DÉFRAGMENTATION DU SYSTÈME DE FICHIERS GFS2

Bien qu'il n'existe pas d'outil de défragmentation pour GFS2 sur Red Hat Enterprise Linux, vous pouvez défragmenter des fichiers individuels en les identifiant avec l'outil **filefrag**, en les copiant dans des fichiers temporaires et en renommant les fichiers temporaires pour remplacer les originaux.

6.2. VERROUILLAGE DES NŒUDS GFS2

Afin d'obtenir les meilleures performances d'un système de fichiers GFS2, il est important de comprendre la théorie de base de son fonctionnement. Un système de fichiers à nœud unique est mis en œuvre avec un cache, dont le but est d'éliminer la latence des accès au disque lors de l'utilisation de données fréquemment demandées. Sous Linux, le cache de page (et historiquement le cache tampon) assure cette fonction de mise en cache.

Avec GFS2, chaque nœud possède son propre cache de pages qui peut contenir une partie des données sur disque. GFS2 utilise un mécanisme de verrouillage appelé *glocks* (prononcé gee-locks) pour maintenir l'intégrité du cache entre les nœuds. Le sous-système glock fournit une fonction de gestion du cache qui est mise en œuvre en utilisant *distributed lock manager* (DLM) comme couche de communication sous-jacente.

Les verrous assurent la protection du cache par inode, de sorte qu'il existe un verrou par inode utilisé pour contrôler la couche de cache. Si ce verrou est accordé en mode partagé (DLM lock mode : PR), les données sous ce verrou peuvent être mises en cache sur un ou plusieurs nœuds en même temps, de sorte que tous les nœuds peuvent avoir un accès local aux données.

Si le glock est accordé en mode exclusif (DLM lock mode : EX), seul un nœud unique peut mettre en cache les données sous ce glock. Ce mode est utilisé par toutes les opérations qui modifient les données (comme l'appel système **write**).

Si un autre nœud demande un verrou qui ne peut être accordé immédiatement, le DLM envoie un message au(x) nœud(s) qui détient(nt) actuellement les verrous bloquant la nouvelle demande pour leur demander d'abandonner leurs verrous. L'abandon des verrous peut être un processus long (par rapport à la plupart des opérations du système de fichiers). L'abandon d'un verrou partagé ne nécessite que l'invalidation du cache, ce qui est relativement rapide et proportionnel à la quantité de données mises en cache.

L'abandon d'un bloc exclusif nécessite une vidange du journal et la réécriture de toutes les données modifiées sur le disque, suivies de l'invalidation conformément au bloc partagé.

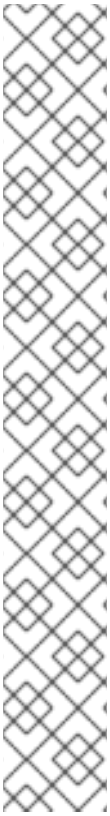
La différence entre un système de fichiers à nœud unique et GFS2 réside donc dans le fait qu'un système de fichiers à nœud unique dispose d'un seul cache et que GFS2 dispose d'un cache séparé sur chaque nœud. Dans les deux cas, le temps de latence pour accéder aux données mises en cache est du même ordre de grandeur, mais le temps de latence pour accéder aux données non mises en cache est beaucoup plus important dans GFS2 si un autre nœud a précédemment mis en cache ces mêmes données.

Les opérations telles que **read** (buffered), **stat,** et **readdir** ne nécessitent qu'un sas partagé. Les opérations telles que **write** (buffered), **mkdir**, **rmdir**, et **unlink** nécessitent un sas exclusif. Les opérations de lecture/écriture d'E/S directes nécessitent un sas différé si aucune allocation n'a lieu, ou un sas exclusif si l'écriture nécessite une allocation (c'est-à-dire l'extension du fichier ou le remplissage de trous).

Il en découle deux considérations principales en matière de performances. Tout d'abord, les opérations en lecture seule se parallélisent très bien au sein d'un cluster, puisqu'elles peuvent être exécutées indépendamment sur chaque nœud. Deuxièmement, les opérations nécessitant un bloc exclusif peuvent réduire les performances si plusieurs nœuds se disputent l'accès au(x) même(s) inode(s). La prise en compte de l'ensemble de travail sur chaque nœud est donc un facteur important dans les performances du système de fichiers GFS2, par exemple lorsque vous effectuez une sauvegarde du système de fichiers, comme décrit dans Sauvegarde d'un système de fichiers GFS2 .

En conséquence, nous recommandons l'utilisation de l'option de montage **noatime** ou **nodiratime** avec GFS2 dans la mesure du possible, avec une préférence pour **noatime** lorsque l'application le permet. Cela empêche les lectures de nécessiter des verrous exclusifs pour mettre à jour l'horodatage de **atime**.

Pour les utilisateurs soucieux de l'efficacité de l'ensemble de travail ou de la mise en cache, GFS2 fournit des outils qui permettent de surveiller les performances d'un système de fichiers GFS2 : Performance Co-Pilot et GFS2 tracepoints.



NOTE

En raison de la manière dont la mise en cache de GFS2 est mise en œuvre, les meilleures performances sont obtenues lorsque l'une ou l'autre des situations suivantes se produit :

- Un inode est utilisé en lecture seule sur tous les nœuds.
- Un inode est écrit ou modifié à partir d'un seul nœud.

Notez que l'insertion et la suppression d'entrées d'un répertoire lors de la création et de la suppression de fichiers est considérée comme une écriture sur l'inode du répertoire.

Il est possible d'enfreindre cette règle à condition de le faire relativement rarement. Ignorer cette règle trop souvent se traduira par une grave pénalité en termes de performances.

Si vous **mmap**() un fichier sur GFS2 avec une correspondance lecture/écriture, mais que vous ne faites que lire à partir de ce fichier, cela ne compte que comme une lecture.

Si vous ne définissez pas le paramètre **noatime mount** , les lectures entraîneront également des écritures pour mettre à jour les horodatages des fichiers. Nous recommandons à tous les utilisateurs de GFS2 d'effectuer le montage avec **noatime**, à moins qu'ils n'aient besoin de **atime**.

6.3. PROBLÈMES LIÉS AU VERROUILLAGE POSIX

Lors de l'utilisation du verrouillage Posix, il convient de tenir compte des éléments suivants :

- L'utilisation de Flocks permet un traitement plus rapide que l'utilisation de verrous Posix.
- Les programmes utilisant des verrous Posix dans GFS2 doivent éviter d'utiliser la fonction **GETLK** car, dans un environnement en grappe, l'ID du processus peut correspondre à un nœud différent de la grappe.

6.4. OPTIMISATION DES PERFORMANCES AVEC GFS2

Il est généralement possible de modifier la manière dont une application gênante stocke ses données afin d'obtenir un avantage considérable en termes de performances.

Un exemple typique d'application problématique est un serveur de courrier électronique. Ceux-ci sont souvent organisés avec un répertoire spool contenant des fichiers pour chaque utilisateur (**mbox**), ou avec un répertoire pour chaque utilisateur contenant un fichier pour chaque message (**maildir**). Lorsque les demandes arrivent par IMAP, l'idéal est de donner à chaque utilisateur une affinité avec un nœud particulier. De cette manière, les demandes de consultation et de suppression de messages électroniques seront généralement traitées à partir du cache de ce nœud. Évidemment, si ce nœud tombe en panne, la session peut être redémarrée sur un autre nœud.

Lorsque le courrier arrive par SMTP, les nœuds individuels peuvent être configurés de manière à transmettre par défaut le courrier d'un utilisateur donné à un nœud particulier. Si le nœud par défaut n'est pas en place, le message peut être enregistré directement dans le spool de l'utilisateur par le nœud de réception. Là encore, cette conception vise à conserver des ensembles particuliers de fichiers en cache sur un seul nœud dans le cas normal, mais à permettre un accès direct en cas de défaillance d'un nœud.

Cette configuration permet d'utiliser au mieux le cache de pages de GFS2 et rend les défaillances transparentes pour l'application, qu'il s'agisse de **imap** ou de **smtp**.

La sauvegarde est souvent un autre point délicat. Encore une fois, si cela est possible, il est largement préférable de sauvegarder l'ensemble de travail de chaque nœud directement à partir du nœud qui met en cache cet ensemble particulier d'inodes. Si vous avez un script de sauvegarde qui s'exécute à intervalles réguliers et qui semble coïncider avec un pic dans le temps de réponse d'une application fonctionnant sur GFS2, il y a de fortes chances que le cluster n'utilise pas le cache de pages de la manière la plus efficace possible.

Évidemment, si vous êtes en mesure d'arrêter l'application pour effectuer une sauvegarde, cela ne posera pas de problème. En revanche, si une sauvegarde est exécutée à partir d'un seul nœud, une fois qu'elle est terminée, une grande partie du système de fichiers sera mise en cache sur ce nœud, ce qui pénalisera les performances pour les accès ultérieurs à partir d'autres nœuds. Ce problème peut être atténué dans une certaine mesure en supprimant le cache de page VFS sur le nœud de sauvegarde une fois la sauvegarde terminée à l'aide de la commande suivante :

```
echo -n 3 >/proc/sys/vm/drop_caches
```

Toutefois, cette solution n'est pas aussi bonne que celle qui consiste à s'assurer que l'ensemble de travail de chaque nœud est soit partagé, soit principalement en lecture seule dans la grappe, soit accessible en grande partie à partir d'un seul nœud.

6.5. DÉPANNAGE DES PERFORMANCES DE GFS2 À L'AIDE DU VIDAGE DES VERROUS DE GFS2

Si les performances de votre cluster sont affectées par une utilisation inefficace de la mise en cache GFS2, vous pouvez observer des temps d'attente d'E/S importants et croissants. Vous pouvez utiliser les informations de vidage de verrou de GFS2 pour déterminer la cause du problème.

Les informations sur le vidage des verrous GFS2 peuvent être recueillies dans le fichier **debugfs** qui se trouve dans le chemin d'accès suivant, en supposant que **debugfs** soit monté sur **/sys/kernel/debug/**:

```
/sys/kernel/debug/gfs2/fsname/glocks
```

Le contenu du fichier est une série de lignes. Chaque ligne commençant par G : représente un glock, et les lignes suivantes, indentées d'un seul espace, représentent une information relative au glock qui les précède immédiatement dans le fichier.

La meilleure façon d'utiliser le fichier **debugfs** est d'utiliser la commande **cat** pour prendre une copie du contenu complet du fichier (cela peut prendre beaucoup de temps si vous avez une grande quantité de mémoire vive et beaucoup d'inodes en cache) pendant que l'application rencontre des problèmes, puis d'examiner les données résultantes à une date ultérieure.

NOTE

Il peut être utile de faire deux copies du fichier **debugfs**, l'une quelques secondes, voire une minute ou deux après l'autre. En comparant les informations du détenteur dans les deux traces relatives au même numéro de glock, vous pouvez dire si la charge de travail progresse (elle est juste lente) ou si elle est bloquée (ce qui est toujours un bogue et devrait être signalé immédiatement à l'assistance de Red Hat).

Les lignes du fichier **debugfs** commençant par H : (holders) représentent des demandes de verrouillage accordées ou en attente d'être accordées. Le champ flags de la ligne f : (holders) indique lesquelles : Le drapeau "W" correspond à une demande en attente, le drapeau "H" à une demande accordée. Les glocks qui présentent un grand nombre de demandes en attente sont probablement ceux qui font l'objet d'une contestation particulière.

Les tableaux suivants indiquent la signification des drapeaux de glock et des drapeaux de porte-glock.

Tableau 6.1. Drapeaux Glock

Drapeau	Nom	Signification
b	Blocage	Valide lorsque l'indicateur de verrouillage est activé et indique que l'opération demandée au DLM peut être bloquée. Cet indicateur est désactivé pour les opérations de rétrogradation et pour les verrous "try". L'objectif de cet indicateur est de permettre la collecte de statistiques sur le temps de réponse du DLM, indépendamment du temps nécessaire aux autres nœuds pour rétrograder les verrous.
d	Rétrogradation en cours	Une demande de rétrogradation différée (à distance)
D	Rétrograder	Une demande de rétrogradation (locale ou à distance)
f	Rinçage des grumes	Le journal doit être engagé avant de sortir ce glock

Drapeau	Nom	Signification
F	Congelé	Les réponses des nœuds distants sont ignorées - la récupération est en cours. Cet indicateur n'est pas lié au gel du système de fichiers, qui utilise un mécanisme différent, mais n'est utilisé que pour la récupération.
i	Invalidation en cours	En cours d'invalidation des pages sous ce glock
I	Initiale	Fixé lorsque la serrure DLM est associée à ce glock
l	Verrouillé	Le glock est en train de changer d'état
L	LRU	Défini lorsque le glock est sur la liste LRU
o	Objet	Défini lorsque le glock est associé à un objet (c'est-à-dire un inode pour les glocks de type 2 et un groupe de ressources pour les glocks de type 3)
p	Rétrogradation en cours	Le glock est en train de répondre à une demande de rétrogradation
q	En attente	Défini lorsqu'un détenteur est mis en file d'attente pour un glock, et effacé lorsque le glock est tenu, mais qu'il n'y a plus de détenteurs restants. Utilisé dans le cadre de l'algorithme qui calcule le temps de maintien minimum d'un glock.
r	Réponse en attente	La réponse reçue du nœud distant est en attente de traitement
y	Sale	Les données doivent être transférées sur le disque avant la mise en circulation de ce glock

Tableau 6.2. Drapeaux pour porte-blocs

Drapeau	Nom	Signification
a	Asynchrone	N'attendez pas le résultat de l'enquête sur le glock (le résultat de l'enquête sera communiqué plus tard)
A	Tous	Tout mode de verrouillage compatible est acceptable
c	Pas de cache	En cas de déverrouillage, rétrograder immédiatement la serrure DLM
e	Pas d'expiration	Ignorer les demandes ultérieures d'annulation de verrou
E	exactes	Doit disposer d'un mode de verrouillage exact
F	Première	Fixé lorsque le détenteur est le premier à bénéficier de cette serrure
H	Titulaire	Indique que le verrou demandé est accordé
p	Priorité	Enqueue holder at the head of the queue
t	Essayer	Serrure "try" A \N- "try" - "try" - "try" - "try" - "try" - "try
T	Essayer 1CB	Un verrou "try" qui envoie un rappel
W	Attendre	Fixé pendant l'attente de l'achèvement de la demande

Après avoir identifié un glock qui pose problème, l'étape suivante consiste à déterminer à quel inode il se rapporte. Le numéro du glock (n : sur la ligne G :) l'indique. Il est de la forme *type/number* et si *type* est égal à 2, le glock est un inode glock et *number* est un numéro d'inode. Pour retrouver l'inode, vous pouvez alors exécuter **find -inum *number*** où *number* est le numéro d'inode converti en décimal à partir du format hexadécimal du fichier glocks.

AVERTISSEMENT

Si vous exécutez la commande **find** sur un système de fichiers en proie à un conflit de verrouillage, vous risquez d'aggraver le problème. Il est conseillé d'arrêter l'application avant d'exécuter la commande **find** lorsque vous recherchez des inodes contestés.

Le tableau suivant indique la signification des différents types de glock.

Tableau 6.3. Types de Glock

Numéro de type	Type de serrure	Utilisation
1	Trans	Verrouillage des transactions
2	Inode	Métadonnées et données des inodes
3	Rgrp	Métadonnées du groupe de ressources
4	Méta	Le superbloc
5	Iopen	Dernière détection rapprochée de l'inode
6	Troupeau	**flock**(2) syscall
8	Quota	Opérations de quotas
9	Journal	Journal mutex

Si le glock identifié était d'un type différent, il est plus probable qu'il soit de type 3 : (groupe de ressources). Si vous observez un nombre significatif de processus en attente d'autres types de glock sous des charges normales, signalez-le à l'assistance de Red Hat.

Si vous voyez un certain nombre de demandes en attente sur un verrou de groupe de ressources, il peut y avoir plusieurs raisons à cela. L'une d'elles est qu'il y a un grand nombre de nœuds par rapport au nombre de groupes de ressources dans le système de fichiers. Une autre raison est que le système de fichiers est presque plein (ce qui nécessite, en moyenne, des recherches plus longues pour les blocs libres). Dans les deux cas, la situation peut être améliorée en ajoutant de l'espace de stockage et en utilisant la commande **gfs2_grow** pour étendre le système de fichiers.

6.6. ACTIVATION DE LA JOURNALISATION DES DONNÉES

En règle générale, GFS2 n'écrit que les métadonnées dans son journal. Le contenu des fichiers est ensuite écrit sur le disque par la synchronisation périodique du noyau qui vide les tampons du système

de fichiers. Un appel à **fsync()** sur un fichier entraîne l'écriture immédiate des données du fichier sur le disque. L'appel est renvoyé lorsque le disque signale que toutes les données ont été écrites en toute sécurité.

La journalisation des données peut se traduire par une réduction du temps **fsync()** pour les très petits fichiers, car les données du fichier sont écrites dans le journal en plus des métadonnées. Cet avantage diminue rapidement à mesure que la taille du fichier augmente. L'écriture sur des fichiers de taille moyenne ou supérieure sera beaucoup plus lente si la journalisation des données est activée.

Les applications qui s'appuient sur **fsync()** pour synchroniser les données des fichiers peuvent voir leurs performances améliorées par l'utilisation de la journalisation des données. La journalisation des données peut être activée automatiquement pour tous les fichiers GFS2 créés dans un répertoire marqué (et tous ses sous-répertoires). La journalisation des données peut également être activée ou désactivée pour les fichiers existants de longueur nulle.

L'activation de la journalisation des données dans un répertoire lui confère la valeur "inherit jdata", ce qui indique que tous les fichiers et répertoires créés ultérieurement dans ce répertoire sont journalisés. Vous pouvez activer et désactiver la journalisation des données sur un fichier à l'aide de la commande **chattr**.

Les commandes suivantes activent la journalisation des données sur le fichier **/mnt/gfs2/gfs2_dir/newfile** et vérifient ensuite si l'indicateur a été correctement défini.

```
# chattr +j /mnt/gfs2/gfs2_dir/newfile
# lsattr /mnt/gfs2/gfs2_dir
---------j--- /mnt/gfs2/gfs2_dir/newfile
```

Les commandes suivantes désactivent la journalisation des données sur le fichier **/mnt/gfs2/gfs2_dir/newfile** et vérifient ensuite si l'indicateur a été correctement défini.

```
# chattr -j /mnt/gfs2/gfs2_dir/newfile
# lsattr /mnt/gfs2/gfs2_dir
---------- /mnt/gfs2/gfs2_dir/newfile
```

Vous pouvez également utiliser la commande **chattr** pour activer l'indicateur **j** sur un répertoire. Lorsque vous activez cet indicateur pour un répertoire, tous les fichiers et répertoires créés ultérieurement dans ce répertoire sont journalisés. Le jeu de commandes suivant définit l'indicateur **j** pour le répertoire **gfs2_dir**, puis vérifie si l'indicateur a été correctement défini. Ensuite, les commandes créent un nouveau fichier appelé **newfile** dans le répertoire **/mnt/gfs2/gfs2_dir**, puis vérifient si l'indicateur **j** a été défini pour le fichier. Puisque l'indicateur **j** est activé pour le répertoire, la journalisation doit également être activée pour **newfile**.

```
# chattr -j /mnt/gfs2/gfs2_dir
# lsattr /mnt/gfs2
---------j--- /mnt/gfs2/gfs2_dir
# touch /mnt/gfs2/gfs2_dir/newfile
# lsattr /mnt/gfs2/gfs2_dir
---------j--- /mnt/gfs2/gfs2_dir/newfile
```

CHAPITRE 7. DIAGNOSTIQUER ET CORRIGER LES PROBLÈMES LIÉS AUX SYSTÈMES DE FICHIERS GFS2

Les procédures suivantes décrivent certains problèmes courants liés à GFS2 et fournissent des informations sur la manière de les résoudre.

7.1. SYSTÈME DE FICHIERS GFS2 INDISPONIBLE POUR UN NŒUD (FONCTION DE RETRAIT DE GFS2)

La fonction GFS2 *withdraw* est une fonction d'intégrité des données du système de fichiers GFS2 qui empêche tout dommage potentiel du système de fichiers dû à un matériel ou à un logiciel noyau défectueux. Si le module du noyau GFS2 détecte une incohérence lors de l'utilisation d'un système de fichiers GFS2 sur un nœud de cluster donné, il se retire du système de fichiers, le rendant indisponible pour ce nœud jusqu'à ce qu'il soit démonté et remonté (ou que la machine détectant le problème soit redémarrée). Tous les autres systèmes de fichiers GFS2 montés restent pleinement fonctionnels sur ce nœud. (La fonction de retrait de GFS2 est moins grave qu'une panique du noyau, qui entraîne la clôture du nœud)

Les principales catégories d'incohérences pouvant entraîner un retrait de la GFS2 sont les suivantes :

- Erreur de cohérence des inodes
- Erreur de cohérence du groupe de ressources
- Erreur de cohérence du journal
- Erreur de cohérence des métadonnées du nombre magique
- Erreur de cohérence du type de métadonnées

Un exemple d'incohérence susceptible d'entraîner un retrait de GFS2 est un nombre de blocs incorrect pour l'inode d'un fichier. Lorsque GFS2 supprime un fichier, il supprime systématiquement tous les blocs de données et de métadonnées référencés par ce fichier. Il vérifie ensuite le nombre de blocs de l'inode. Si le nombre de blocs n'est pas égal à 1 (ce qui signifie qu'il ne reste que l'inode du disque lui-même), cela indique une incohérence du système de fichiers, car le nombre de blocs de l'inode ne correspond pas aux blocs réellement utilisés pour le fichier.

Dans de nombreux cas, le problème peut avoir été causé par un matériel défectueux (mémoire défectueuse, carte mère, HBA, lecteurs de disques, câbles, etc.) Il peut également être dû à un bogue du noyau (un autre module du noyau écrasant accidentellement la mémoire de GFS2) ou à une détérioration réelle du système de fichiers (causée par un bogue de GFS2).

Dans la plupart des cas, la meilleure façon de récupérer un système de fichiers GFS2 retiré est de redémarrer ou de clôturer le nœud. Le système de fichiers GFS2 retiré vous donnera l'occasion de déplacer les services vers un autre nœud du cluster. Une fois les services déplacés, vous pouvez redémarrer le nœud ou forcer une clôture à l'aide de cette commande.

```
pcs stonith fence node
```

AVERTISSEMENT

N'essayez pas de démonter et de remonter le système de fichiers manuellement avec les commandes **umount** et **mount**. Vous devez utiliser la commande **pcs**, sinon Pacemaker détectera que le service de système de fichiers a disparu et clôturera le nœud.

Le problème de cohérence à l'origine du retrait peut rendre impossible l'arrêt du service de système de fichiers, car il risque de bloquer le système.

Si le problème persiste après un remontage, vous devez arrêter le service de système de fichiers pour démonter le système de fichiers sur tous les nœuds du cluster, puis effectuer un contrôle du système de fichiers à l'aide de la commande **fsck.gfs2** avant de redémarrer le service à l'aide de la procédure suivante.

1. Redémarrer le nœud concerné.
2. Désactivez le service de système de fichiers non clonés dans Pacemaker pour démonter le système de fichiers de chaque nœud du cluster.

```
# pcs resource disable --wait=100 mydata_fs
```

3. À partir d'un nœud de la grappe, exécutez la commande **fsck.gfs2** sur le périphérique du système de fichiers pour vérifier et réparer tout dommage du système de fichiers.

```
# fsck.gfs2 -y /dev/vg_mydata/mydata > /tmp/fsck.out
```

4. Remontez le système de fichiers GFS2 à partir de tous les nœuds en réactivant le service de système de fichiers :

```
# pcs resource enable --wait=100 mydata_fs
```

Vous pouvez remplacer la fonction de retrait de GFS2 en montant le système de fichiers avec l'option **-o errors=panic** spécifiée dans le service du système de fichiers.

```
# pcs resource update mydata_fs “options=noatime,errors=panic”
```

Lorsque cette option est spécifiée, toutes les erreurs qui entraîneraient normalement le retrait du système provoquent une panique du noyau. Les communications du nœud sont alors interrompues, ce qui entraîne la clôture du nœud. Cette option est particulièrement utile pour les grappes qui sont laissées sans surveillance pendant de longues périodes sans contrôle ni intervention.

En interne, la fonction de retrait de GFS2 fonctionne en déconnectant le protocole de verrouillage afin de garantir que toutes les opérations ultérieures du système de fichiers se traduisent par des erreurs d'E/S. Par conséquent, lorsque le retrait se produit, il est normal de voir un certain nombre d'erreurs d'E/S provenant du périphérique de mappage signalées dans les journaux du système.

7.2. LE SYSTÈME DE FICHIERS GFS2 SE BLOQUE ET NÉCESSITE LE REDÉMARRAGE D'UN NŒUD

Si votre système de fichiers GFS2 se bloque et ne renvoie pas les commandes exécutées, mais que le redémarrage d'un nœud spécifique ramène le système à la normale, cela peut indiquer un problème de verrouillage ou un bogue. Si cela se produit, rassemblez les données GFS2 pendant l'une de ces occurrences et ouvrez un ticket d'assistance avec l'assistance de Red Hat, comme décrit dans Rassembler les données GFS2 pour le dépannage .

7.3. LE SYSTÈME DE FICHIERS GFS2 SE BLOQUE ET NÉCESSITE LE REDÉMARRAGE DE TOUS LES NŒUDS

Si votre système de fichiers GFS2 se bloque et ne renvoie pas les commandes exécutées, ce qui vous oblige à redémarrer tous les nœuds de la grappe avant de l'utiliser, vérifiez les points suivants.

- Il se peut que votre clôture ait échoué. Les systèmes de fichiers GFS2 se figent pour garantir l'intégrité des données en cas d'échec de la clôture. Vérifiez les journaux de messages pour voir si des clôtures ont échoué au moment du blocage. Assurez-vous que la clôture est configurée correctement.

- Il se peut que le système de fichiers GFS2 ait été retiré. Recherchez le mot **withdraw** dans les journaux de messages et vérifiez si des messages et des traces d'appel de GFS2 indiquent que le système de fichiers a été retiré. Un retrait est le signe d'une corruption du système de fichiers, d'une panne de stockage ou d'un bogue. Dès qu'il sera possible de démonter le système de fichiers, vous devrez exécuter la procédure suivante :

  a. Redémarrer le nœud sur lequel le retrait s'est produit.

  ```
  # /sbin/reboot
  ```

  b. Arrêtez la ressource du système de fichiers pour démonter le système de fichiers GFS2 sur tous les nœuds.

  ```
  # pcs resource disable --wait=100 mydata_fs
  ```

  c. Capturez les métadonnées à l'aide de la commande **gfs2_edit savemeta…**. Vous devez vous assurer qu'il y a suffisamment d'espace pour le fichier, qui peut être volumineux dans certains cas. Dans cet exemple, les métadonnées sont enregistrées dans un fichier du répertoire **/root**.

  ```
  # gfs2_edit savemeta /dev/vg_mydata/mydata /root/gfs2metadata.gz
  ```

  d. Mettre à jour le paquet **gfs2-utils**.

  ```
  # sudo dnf update gfs2-utils
  ```

  e. Sur un nœud, exécutez la commande **fsck.gfs2** sur le système de fichiers pour en vérifier l'intégrité et réparer les dommages éventuels.

  ```
  # fsck.gfs2 -y /dev/vg_mydata/mydata > /tmp/fsck.out
  ```

  f. Une fois la commande **fsck.gfs2** terminée, réactivez la ressource du système de fichiers pour la remettre en service :

  ```
  # pcs resource enable --wait=100 mydata_fs
  ```

g. Ouvrez un ticket d'assistance auprès de Red Hat Support. Informez-les que vous avez subi un retrait de GFS2 et fournissez les journaux et les informations de débogage générés par les commandes **sosreports** et **gfs2_edit savemeta**.
Dans certains cas de retrait de GFS2, les commandes qui tentent d'accéder au système de fichiers ou à son périphérique de bloc peuvent se bloquer. Dans ce cas, un redémarrage complet est nécessaire pour redémarrer le cluster.

Pour plus d'informations sur la fonction de retrait de GFS2, voir Système de fichiers GFS2 indisponible pour un nœud (fonction de retrait de GFS2).

- Cette erreur peut indiquer un problème de verrouillage ou un bogue. Rassemblez des données lors de l'une de ces occurrences et ouvrez un ticket d'assistance avec l'assistance de Red Hat, comme décrit dans Rassembler des données GFS2 pour le dépannage .

7.4. LE SYSTÈME DE FICHIERS GFS2 NE SE MONTE PAS SUR LE NŒUD DE CLUSTER NOUVELLEMENT AJOUTÉ

Si vous ajoutez un nouveau nœud à un cluster et que vous constatez que vous ne pouvez pas monter votre système de fichiers GFS2 sur ce nœud, il se peut que vous ayez moins de journaux sur le système de fichiers GFS2 que de nœuds tentant d'accéder au système de fichiers GFS2. Vous devez disposer d'un journal par hôte GFS2 sur lequel vous souhaitez monter le système de fichiers (à l'exception des systèmes de fichiers GFS2 montés avec l'option de montage **spectator**, qui ne nécessitent pas de journal). Vous pouvez ajouter des journaux à un système de fichiers GFS2 à l'aide de la commande **gfs2_jadd**, comme décrit dans Ajout de journaux à un système de fichiers GFS2 .

7.5. ESPACE INDIQUÉ COMME UTILISÉ DANS UN SYSTÈME DE FICHIERS VIDE

If you have an empty GFS2 file system, the **df** command will show that there is space being taken up. This is because GFS2 file system journals consume space (number of journals * journal size) on disk. If you created a GFS2 file system with a large number of journals or specified a large journal size then you will be see (number of journals * journal size) as already in use when you execute the **df** command. Even if you did not specify a large number of journals or large journals, small GFS2 file systems (in the 1GB or less range) will show a large amount of space as being in use with the default GFS2 journal size.

7.6. COLLECTE DE DONNÉES GFS2 POUR LE DÉPANNAGE

Si votre système de fichiers GFS2 se bloque et ne renvoie pas les commandes exécutées contre lui et que vous estimez devoir ouvrir un ticket avec l'assistance de Red Hat, vous devez tout d'abord rassembler les données suivantes :

- Le vidage du verrou GFS2 pour le système de fichiers sur chaque nœud :

```
cat /sys/kernel/debug/gfs2/fsname/glocks >glocks.fsname.nodename
```

- Le dump de verrouillage DLM pour le système de fichiers sur chaque nœud : Vous pouvez obtenir ces informations à l'aide de la commande **dlm_tool**:

```
dlm_tool lockdebug -sv lsname
```

Dans cette commande, *lsname* est le nom de l'espace de verrouillage utilisé par DLM pour le système de fichiers en question. Vous trouverez cette valeur dans la sortie de la commande **group_tool**.

- La sortie de la commande **sysrq -t**.
- Le contenu du fichier **/var/log/messages**.

Une fois que vous avez rassemblé ces données, vous pouvez ouvrir un ticket avec le support de Red Hat et fournir les données que vous avez collectées.

CHAPITRE 8. SYSTÈMES DE FICHIERS GFS2 DANS UN CLUSTER

Utilisez les procédures administratives suivantes pour configurer les systèmes de fichiers GFS2 dans un cluster de haute disponibilité Red Hat.

8.1. CONFIGURATION D'UN SYSTÈME DE FICHIERS GFS2 DANS UN CLUSTER

Vous pouvez configurer un cluster Pacemaker comprenant des systèmes de fichiers GFS2 à l'aide de la procédure suivante. Dans cet exemple, vous créez trois systèmes de fichiers GFS2 sur trois volumes logiques dans un cluster à deux nœuds.

Conditions préalables

- Installez et démarrez le logiciel de cluster sur les deux nœuds du cluster et créez un cluster de base à deux nœuds.
- Configurer la clôture pour le cluster.

Pour plus d'informations sur la création d'un cluster Pacemaker et la configuration de la clôture pour le cluster, voir Création d'un cluster Red Hat High-Availability avec Pacemaker .

Procédure

1. Sur les deux nœuds du cluster, activez le référentiel Resilient Storage correspondant à l'architecture de votre système. Par exemple, pour activer le référentiel Resilient Storage pour un système x86_64, vous pouvez entrer la commande **subscription-manager** suivante :

   ```
   # subscription-manager repos --enable=rhel-9-for-x86_64-resilientstorage-rpms
   ```

   Notez que le référentiel de stockage résilient est un surensemble du référentiel de haute disponibilité. Si vous activez le référentiel de stockage résilient, il n'est pas nécessaire d'activer également le référentiel de haute disponibilité.

2. Sur les deux nœuds du cluster, installez les paquets **lvm2-lockd**, **gfs2-utils**, et **dlm**. Pour prendre en charge ces paquets, vous devez être abonné au canal AppStream et au canal Resilient Storage.

   ```
   # dnf install lvm2-lockd gfs2-utils dlm
   ```

3. Sur les deux nœuds du cluster, définissez l'option de configuration **use_lvmlockd** dans le fichier **/etc/lvm/lvm.conf** sur **use_lvmlockd=1**.

   ```
   ...
   use_lvmlockd = 1
   ...
   ```

4. Réglez le paramètre global du stimulateur cardiaque **no-quorum-policy** sur **freeze**.

NOTE

Par défaut, la valeur de **no-quorum-policy** est fixée à **stop**, indiquant qu'une fois le quorum perdu, toutes les ressources sur la partition restante seront immédiatement arrêtées. En général, cette valeur par défaut est l'option la plus sûre et la plus optimale, mais contrairement à la plupart des ressources, GFS2 a besoin du quorum pour fonctionner. Lorsque le quorum est perdu, les applications utilisant les montages GFS2 et le montage GFS2 lui-même ne peuvent pas être arrêtés correctement. Toute tentative d'arrêt de ces ressources sans quorum échouera, ce qui aura pour conséquence de clôturer l'ensemble du cluster à chaque fois que le quorum est perdu.

Pour remédier à cette situation, définissez **no-quorum-policy** sur **freeze** lorsque GFS2 est utilisé. Cela signifie que lorsque le quorum est perdu, la partition restante ne fera rien jusqu'à ce que le quorum soit rétabli.

```
[root@z1 ~]# pcs property set no-quorum-policy=freeze
```

5. Configurer une ressource **dlm**. Il s'agit d'une dépendance nécessaire pour configurer un système de fichiers GFS2 dans un cluster. Cet exemple crée la ressource **dlm** dans le cadre d'un groupe de ressources nommé **locking**.

```
[root@z1 ~]# pcs resource create dlm --group locking ocf:pacemaker:controld op
monitor interval=30s on-fail=fence
```

6. Clonez le groupe de ressources **locking** afin que le groupe de ressources puisse être actif sur les deux nœuds du cluster.

```
[root@z1 ~]# pcs resource clone locking interleave=true
```

7. Créez une ressource **lvmlockd** dans le cadre du groupe de ressources **locking**.

```
[root@z1 ~]# pcs resource create lvmlockd --group locking ocf:heartbeat:lvmlockd op
monitor interval=30s on-fail=fence
```

8. Vérifiez l'état du cluster pour vous assurer que le groupe de ressources **locking** a démarré sur les deux nœuds du cluster.

```
[root@z1 ~]# pcs status --full
Cluster name: my_cluster
[...]

Online: [ z1.example.com (1) z2.example.com (2) ]

Full list of resources:

 smoke-apc      (stonith:fence_apc):    Started z1.example.com
 Clone Set: locking-clone [locking]
     Resource Group: locking:0
         dlm    (ocf::pacemaker:controld):      Started z1.example.com
         lvmlockd       (ocf::heartbeat:lvmlockd):      Started z1.example.com
     Resource Group: locking:1
```

```
    dlm    (ocf::pacemaker:controld):      Started z2.example.com
    lvmlockd       (ocf::heartbeat:lvmlockd):      Started z2.example.com
Started: [ z1.example.com z2.example.com ]
```

9. Sur un nœud de la grappe, créez deux groupes de volumes partagés. Un groupe de volumes contiendra deux systèmes de fichiers GFS2 et l'autre groupe de volumes contiendra un système de fichiers GFS2.

NOTE

Si votre groupe de volumes LVM contient un ou plusieurs volumes physiques résidant sur un stockage en bloc distant, tel qu'une cible iSCSI, Red Hat vous recommande de vous assurer que le service démarre avant le démarrage de Pacemaker. Pour plus d'informations sur la configuration de l'ordre de démarrage d'un volume physique distant utilisé par un cluster Pacemaker, reportez-vous à Configuration de l'ordre de démarrage pour les dépendances de ressources non gérées par Pacemaker.

La commande suivante crée le groupe de volumes partagés **shared_vg1** sur **/dev/vdb**.

```
[root@z1 ~]# vgcreate --shared shared_vg1 /dev/vdb
  Physical volume "/dev/vdb" successfully created.
  Volume group "shared_vg1" successfully created
  VG shared_vg1 starting dlm lockspace
  Starting locking.  Waiting until locks are ready...
```

La commande suivante crée le groupe de volumes partagés **shared_vg2** sur **/dev/vdc**.

```
[root@z1 ~]# vgcreate --shared shared_vg2 /dev/vdc
  Physical volume "/dev/vdc" successfully created.
  Volume group "shared_vg2" successfully created
  VG shared_vg2 starting dlm lockspace
  Starting locking.  Waiting until locks are ready...
```

10. Sur le deuxième nœud du cluster :

    a. Ajoutez les périphériques partagés au fichier des périphériques LVM.

    ```
    [root@z2 ~]# lvmdevices --adddev /dev/vdb
    [root@z2 ~]# lvmdevices --adddev /dev/vdc
    ```

    b. Lancez le gestionnaire de verrous pour chacun des groupes de volumes partagés.

    ```
    [root@z2 ~]# vgchange --lockstart shared_vg1
      VG shared_vg1 starting dlm lockspace
      Starting locking.  Waiting until locks are ready...
    [root@z2 ~]# vgchange --lockstart shared_vg2
      VG shared_vg2 starting dlm lockspace
      Starting locking.  Waiting until locks are ready...
    ```

11. Sur un nœud de la grappe, créez les volumes logiques partagés et formatez les volumes avec un système de fichiers GFS2. Un journal est nécessaire pour chaque nœud qui monte le système de fichiers. Veillez à créer suffisamment de journaux pour chacun des nœuds de votre grappe.

Le format du nom de la table de verrouillage est *ClusterName:FSName*, où *ClusterName* est le nom de la grappe pour laquelle le système de fichiers GFS2 est créé et *FSName* est le nom du système de fichiers, qui doit être unique pour tous les systèmes de fichiers **lock_dlm** de la grappe.

```
[root@z1 ~]# lvcreate --activate sy -L5G -n shared_lv1 shared_vg1
  Logical volume "shared_lv1" created.
[root@z1 ~]# lvcreate --activate sy -L5G -n shared_lv2 shared_vg1
  Logical volume "shared_lv2" created.
[root@z1 ~]# lvcreate --activate sy -L5G -n shared_lv1 shared_vg2
  Logical volume "shared_lv1" created.

[root@z1 ~]# mkfs.gfs2 -j2 -p lock_dlm -t my_cluster:gfs2-demo1
/dev/shared_vg1/shared_lv1
[root@z1 ~]# mkfs.gfs2 -j2 -p lock_dlm -t my_cluster:gfs2-demo2
/dev/shared_vg1/shared_lv2
[root@z1 ~]# mkfs.gfs2 -j2 -p lock_dlm -t my_cluster:gfs2-demo3
/dev/shared_vg2/shared_lv1
```

12. Créez une ressource **LVM-activate** pour chaque volume logique afin d'activer automatiquement ce volume logique sur tous les nœuds.

    a. Créez une ressource **LVM-activate** nommée **sharedlv1** pour le volume logique **shared_lv1** dans le groupe de volumes **shared_vg1**. Cette commande crée également le groupe de ressources **shared_vg1** qui inclut la ressource. Dans cet exemple, le groupe de ressources porte le même nom que le groupe de volumes partagés qui comprend le volume logique.

    ```
    [root@z1 ~]# pcs resource create sharedlv1 --group shared_vg1 ocf:heartbeat:LVM-
    activate lvname=shared_lv1 vgname=shared_vg1 activation_mode=shared
    vg_access_mode=lvmlockd
    ```

    b. Créez une ressource **LVM-activate** nommée **sharedlv2** pour le volume logique **shared_lv2** dans le groupe de volumes **shared_vg1**. Cette ressource fera également partie du groupe de ressources **shared_vg1**.

    ```
    [root@z1 ~]# pcs resource create sharedlv2 --group shared_vg1 ocf:heartbeat:LVM-
    activate lvname=shared_lv2 vgname=shared_vg1 activation_mode=shared
    vg_access_mode=lvmlockd
    ```

    c. Créez une ressource **LVM-activate** nommée **sharedlv3** pour le volume logique **shared_lv1** dans le groupe de volumes **shared_vg2**. Cette commande crée également le groupe de ressources **shared_vg2** qui inclut la ressource.

    ```
    [root@z1 ~]# pcs resource create sharedlv3 --group shared_vg2 ocf:heartbeat:LVM-
    activate lvname=shared_lv1 vgname=shared_vg2 activation_mode=shared
    vg_access_mode=lvmlockd
    ```

13. Clonez les deux nouveaux groupes de ressources.

```
[root@z1 ~]# pcs resource clone shared_vg1 interleave=true
[root@z1 ~]# pcs resource clone shared_vg2 interleave=true
```

14. Configurez les contraintes d'ordre pour vous assurer que le groupe de ressources **locking** qui inclut les ressources **dlm** et **lvmlockd** démarre en premier.

```
[root@z1 ~]# pcs constraint order start locking-clone then shared_vg1-clone
Adding locking-clone shared_vg1-clone (kind: Mandatory) (Options: first-action=start then-action=start)
[root@z1 ~]# pcs constraint order start locking-clone then shared_vg2-clone
Adding locking-clone shared_vg2-clone (kind: Mandatory) (Options: first-action=start then-action=start)
```

15. Configurez les contraintes de colocation pour vous assurer que les groupes de ressources **vg1** et **vg2** démarrent sur le même nœud que le groupe de ressources **locking**.

```
[root@z1 ~]# pcs constraint colocation add shared_vg1-clone with locking-clone
[root@z1 ~]# pcs constraint colocation add shared_vg2-clone with locking-clone
```

16. Sur les deux nœuds du cluster, vérifiez que les volumes logiques sont actifs. Il peut y avoir un délai de quelques secondes.

```
[root@z1 ~]# lvs
  LV         VG         Attr       LSize
  shared_lv1 shared_vg1 -wi-a----- 5.00g
  shared_lv2 shared_vg1 -wi-a----- 5.00g
  shared_lv1 shared_vg2 -wi-a----- 5.00g

[root@z2 ~]# lvs
  LV         VG         Attr       LSize
  shared_lv1 shared_vg1 -wi-a----- 5.00g
  shared_lv2 shared_vg1 -wi-a----- 5.00g
  shared_lv1 shared_vg2 -wi-a----- 5.00g
```

17. Créez une ressource de système de fichiers pour monter automatiquement chaque système de fichiers GFS2 sur tous les nœuds.
    Vous ne devez pas ajouter le système de fichiers au fichier **/etc/fstab** car il sera géré comme une ressource de cluster Pacemaker. Les options de montage peuvent être spécifiées dans le cadre de la configuration de la ressource à l'aide de la commande **options=*options***. Exécutez la commande **pcs resource describe Filesystem** pour afficher les options de configuration complètes.

    Les commandes suivantes créent les ressources du système de fichiers. Elles ajoutent chaque ressource au groupe de ressources qui comprend la ressource de volume logique pour ce système de fichiers.

```
[root@z1 ~]# pcs resource create sharedfs1 --group shared_vg1 ocf:heartbeat:Filesystem device="/dev/shared_vg1/shared_lv1" directory="/mnt/gfs1" fstype="gfs2" options=noatime op monitor interval=10s on-fail=fence
[root@z1 ~]# pcs resource create sharedfs2 --group shared_vg1 ocf:heartbeat:Filesystem device="/dev/shared_vg1/shared_lv2" directory="/mnt/gfs2" fstype="gfs2" options=noatime op monitor interval=10s on-fail=fence
[root@z1 ~]# pcs resource create sharedfs3 --group shared_vg2 ocf:heartbeat:Filesystem device="/dev/shared_vg2/shared_lv1" directory="/mnt/gfs3" fstype="gfs2" options=noatime op monitor interval=10s on-fail=fence
```

Verification steps

1. Vérifiez que les systèmes de fichiers GFS2 sont montés sur les deux nœuds du cluster.

```
[root@z1 ~]# mount | grep gfs2
/dev/mapper/shared_vg1-shared_lv1 on /mnt/gfs1 type gfs2 (rw,noatime,seclabel)
/dev/mapper/shared_vg1-shared_lv2 on /mnt/gfs2 type gfs2 (rw,noatime,seclabel)
/dev/mapper/shared_vg2-shared_lv1 on /mnt/gfs3 type gfs2 (rw,noatime,seclabel)

[root@z2 ~]# mount | grep gfs2
/dev/mapper/shared_vg1-shared_lv1 on /mnt/gfs1 type gfs2 (rw,noatime,seclabel)
/dev/mapper/shared_vg1-shared_lv2 on /mnt/gfs2 type gfs2 (rw,noatime,seclabel)
/dev/mapper/shared_vg2-shared_lv1 on /mnt/gfs3 type gfs2 (rw,noatime,seclabel)
```

2. Vérifier l'état de la grappe.

```
[root@z1 ~]# pcs status --full
Cluster name: my_cluster
[...]

Full list of resources:

 smoke-apc      (stonith:fence_apc):    Started z1.example.com
 Clone Set: locking-clone [locking]
     Resource Group: locking:0
         dlm    (ocf::pacemaker:controld):      Started z2.example.com
         lvmlockd       (ocf::heartbeat:lvmlockd):      Started z2.example.com
     Resource Group: locking:1
         dlm    (ocf::pacemaker:controld):      Started z1.example.com
         lvmlockd       (ocf::heartbeat:lvmlockd):      Started z1.example.com
     Started: [ z1.example.com z2.example.com ]
 Clone Set: shared_vg1-clone [shared_vg1]
     Resource Group: shared_vg1:0
         sharedlv1      (ocf::heartbeat:LVM-activate):  Started z2.example.com
         sharedlv2      (ocf::heartbeat:LVM-activate):  Started z2.example.com
         sharedfs1      (ocf::heartbeat:Filesystem):    Started z2.example.com
         sharedfs2      (ocf::heartbeat:Filesystem):    Started z2.example.com
     Resource Group: shared_vg1:1
         sharedlv1      (ocf::heartbeat:LVM-activate):  Started z1.example.com
         sharedlv2      (ocf::heartbeat:LVM-activate):  Started z1.example.com
         sharedfs1      (ocf::heartbeat:Filesystem):    Started z1.example.com
         sharedfs2      (ocf::heartbeat:Filesystem):    Started z1.example.com
     Started: [ z1.example.com z2.example.com ]
 Clone Set: shared_vg2-clone [shared_vg2]
     Resource Group: shared_vg2:0
         sharedlv3      (ocf::heartbeat:LVM-activate):  Started z2.example.com
         sharedfs3      (ocf::heartbeat:Filesystem):    Started z2.example.com
     Resource Group: shared_vg2:1
         sharedlv3      (ocf::heartbeat:LVM-activate):  Started z1.example.com
         sharedfs3      (ocf::heartbeat:Filesystem):    Started z1.example.com
     Started: [ z1.example.com z2.example.com ]

...
```

Ressources supplémentaires

- Configuration des systèmes de fichiers GFS2
- Configurer un cluster Red Hat High Availability sur Microsoft Azure

- Configuration d'un cluster Red Hat High Availability sur AWS
- Configuration de Red Hat High Availability Cluster sur Google Cloud Platform
- Configurer le stockage en bloc partagé pour un cluster Red Hat High Availability sur Alibaba Cloud

8.2. CONFIGURATION D'UN SYSTÈME DE FICHIERS GFS2 CRYPTÉ DANS UN CLUSTER

Vous pouvez créer un cluster Pacemaker qui inclut un système de fichiers GFS2 crypté LUKS à l'aide de la procédure suivante. Dans cet exemple, vous créez un système de fichiers GFS2 sur un volume logique et vous chiffrez le système de fichiers. Les systèmes de fichiers GFS2 chiffrés sont pris en charge par l'agent de ressources **crypt**, qui prend en charge le chiffrement LUKS.

Cette procédure comporte trois parties :

- Configuration d'un volume logique partagé dans un cluster Pacemaker
- Chiffrement du volume logique et création d'une ressource **crypt**
- Formatage du volume logique crypté avec un système de fichiers GFS2 et création d'une ressource de système de fichiers pour le cluster

8.2.1. Configurer un volume logique partagé dans un cluster Pacemaker

Conditions préalables

- Installez et démarrez le logiciel de cluster sur deux nœuds de cluster et créez un cluster de base à deux nœuds.
- Configurer la clôture pour le cluster.

Pour plus d'informations sur la création d'un cluster Pacemaker et la configuration de la clôture pour le cluster, voir Création d'un cluster Red Hat High-Availability avec Pacemaker .

Procédure

1. Sur les deux nœuds du cluster, activez le référentiel Resilient Storage correspondant à l'architecture de votre système. Par exemple, pour activer le référentiel Resilient Storage pour un système x86_64, vous pouvez entrer la commande **subscription-manager** suivante :

   ```
   # subscription-manager repos --enable=rhel-9-for-x86_64-resilientstorage-rpms
   ```

   Notez que le référentiel de stockage résilient est un surensemble du référentiel de haute disponibilité. Si vous activez le référentiel de stockage résilient, il n'est pas nécessaire d'activer également le référentiel de haute disponibilité.

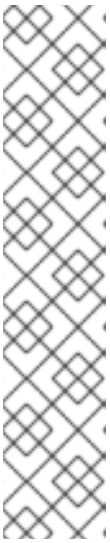
2. Sur les deux nœuds du cluster, installez les paquets **lvm2-lockd**, **gfs2-utils**, et **dlm**. Pour prendre en charge ces paquets, vous devez être abonné au canal AppStream et au canal Resilient Storage.

   ```
   # dnf install lvm2-lockd gfs2-utils dlm
   ```

3. Sur les deux nœuds du cluster, définissez l'option de configuration **use_lvmlockd** dans le fichier **/etc/lvm/lvm.conf** sur **use_lvmlockd=1**.

```
...
use_lvmlockd = 1
...
```

4. Réglez le paramètre global du stimulateur cardiaque **no-quorum-policy** sur **freeze**.



NOTE

Par défaut, la valeur de **no-quorum-policy** est fixée à **stop**, indiquant que lorsque le quorum est perdu, toutes les ressources sur la partition restante seront immédiatement arrêtées. En général, cette valeur par défaut est l'option la plus sûre et la plus optimale, mais contrairement à la plupart des ressources, GFS2 a besoin du quorum pour fonctionner. Lorsque le quorum est perdu, les applications utilisant les montages GFS2 et le montage GFS2 lui-même ne peuvent pas être arrêtés correctement. Toute tentative d'arrêt de ces ressources sans quorum échouera, ce qui aura pour conséquence de clôturer l'ensemble du cluster à chaque fois que le quorum est perdu.

Pour remédier à cette situation, définissez **no-quorum-policy** sur **freeze** lorsque GFS2 est utilisé. Cela signifie que lorsque le quorum est perdu, la partition restante ne fera rien jusqu'à ce que le quorum soit rétabli.

```
[root@z1 ~]# pcs property set no-quorum-policy=freeze
```

5. Configurer une ressource **dlm**. Il s'agit d'une dépendance nécessaire pour configurer un système de fichiers GFS2 dans un cluster. Cet exemple crée la ressource **dlm** dans le cadre d'un groupe de ressources nommé **locking**.

```
[root@z1 ~]# pcs resource create dlm --group locking ocf:pacemaker:controld op
monitor interval=30s on-fail=fence
```

6. Clonez le groupe de ressources **locking** afin que le groupe de ressources puisse être actif sur les deux nœuds du cluster.

```
[root@z1 ~]# pcs resource clone locking interleave=true
```

7. Créez une ressource **lvmlockd** dans le cadre du groupe **locking**.

```
[root@z1 ~]# pcs resource create lvmlockd --group locking ocf:heartbeat:lvmlockd op
monitor interval=30s on-fail=fence
```

8. Vérifiez l'état du cluster pour vous assurer que le groupe de ressources **locking** a démarré sur les deux nœuds du cluster.

```
[root@z1 ~]# pcs status --full
Cluster name: my_cluster
[...]

Online: [ z1.example.com (1) z2.example.com (2) ]
```

```
Full list of resources:

 smoke-apc      (stonith:fence_apc):    Started z1.example.com
 Clone Set: locking-clone [locking]
     Resource Group: locking:0
         dlm    (ocf::pacemaker:controld):      Started z1.example.com
         lvmlockd       (ocf::heartbeat:lvmlockd):      Started z1.example.com
     Resource Group: locking:1
         dlm    (ocf::pacemaker:controld):      Started z2.example.com
         lvmlockd       (ocf::heartbeat:lvmlockd):      Started z2.example.com
     Started: [ z1.example.com z2.example.com ]
```

9. Sur un nœud de la grappe, créez un groupe de volumes partagés.

NOTE

Si votre groupe de volumes LVM contient un ou plusieurs volumes physiques résidant sur un stockage en bloc distant, tel qu'une cible iSCSI, Red Hat vous recommande de vous assurer que le service démarre avant le démarrage de Pacemaker. Pour obtenir des informations sur la configuration de l'ordre de démarrage d'un volume physique distant utilisé par un cluster Pacemaker, reportez-vous à Configuration de l'ordre de démarrage pour les dépendances de ressources non gérées par Pacemaker.

La commande suivante crée le groupe de volumes partagés **shared_vg1** sur **/dev/sda1**.

```
[root@z1 ~]# vgcreate --shared shared_vg1 /dev/sda1
  Physical volume "/dev/sda1" successfully created.
  Volume group "shared_vg1" successfully created
  VG shared_vg1 starting dlm lockspace
  Starting locking.  Waiting until locks are ready...
```

10. Sur le deuxième nœud du cluster :

    a. Ajoutez le périphérique partagé au fichier des périphériques LVM.

    ```
    [root@z2 ~]# lvmdevices --adddev /dev/sda1
    ```

    b. Démarrer le gestionnaire de verrouillage pour le groupe de volumes partagés.

    ```
    [root@z2 ~]# vgchange --lockstart shared_vg1
      VG shared_vg1 starting dlm lockspace
      Starting locking.  Waiting until locks are ready...
    ```

11. Sur un nœud de la grappe, créez le volume logique partagé.

```
[root@z1 ~]# lvcreate --activate sy -L5G -n shared_lv1 shared_vg1
  Logical volume "shared_lv1" created.
```

12. Créez une ressource **LVM-activate** pour le volume logique afin d'activer automatiquement le volume logique sur tous les nœuds.
La commande suivante crée une ressource **LVM-activate** nommée **sharedlv1** pour le volume logique **shared_lv1** dans le groupe de volumes **shared_vg1**. Cette commande crée également

le groupe de ressources **shared_vg1** qui inclut la ressource. Dans cet exemple, le groupe de ressources porte le même nom que le groupe de volumes partagés qui comprend le volume logique.

```
[root@z1 ~]# pcs resource create sharedlv1 --group shared_vg1 ocf:heartbeat:LVM-activate lvname=shared_lv1 vgname=shared_vg1 activation_mode=shared vg_access_mode=lvmlockd
```

13. Cloner le nouveau groupe de ressources.

```
[root@z1 ~]# pcs resource clone shared_vg1 interleave=true
```

14. Configurez une contrainte d'ordre pour garantir que le groupe de ressources **locking** qui inclut les ressources **dlm** et **lvmlockd** démarre en premier.

```
[root@z1 ~]# pcs constraint order start locking-clone then shared_vg1-clone
Adding locking-clone shared_vg1-clone (kind: Mandatory) (Options: first-action=start then-action=start)
```

15. Configurer les contraintes de colocation pour s'assurer que les groupes de ressources **vg1** et **vg2** démarrent sur le même nœud que le groupe de ressources **locking**.

```
[root@z1 ~]# pcs constraint colocation add shared_vg1-clone with locking-clone
```

Verification steps

Sur les deux nœuds du cluster, vérifiez que le volume logique est actif. Il peut y avoir un délai de quelques secondes.

```
[root@z1 ~]# lvs
  LV        VG         Attr      LSize
  shared_lv1 shared_vg1  -wi-a----- 5.00g

[root@z2 ~]# lvs
  LV        VG         Attr      LSize
  shared_lv1 shared_vg1  -wi-a----- 5.00g
```

8.2.2. Cryptage du volume logique et création d'une ressource cryptée

Conditions préalables

- Vous avez configuré un volume logique partagé dans un cluster Pacemaker.

Procédure

1. Sur un nœud du cluster, créez un nouveau fichier qui contiendra la clé cryptographique et définissez les autorisations sur le fichier de sorte qu'il ne soit lisible que par root.

```
[root@z1 ~]# touch /etc/crypt_keyfile
[root@z1 ~]# chmod 600 /etc/crypt_keyfile
```

2. Créer la clé cryptographique.

```
[root@z1 ~]# dd if=/dev/urandom bs=4K count=1 of=/etc/crypt_keyfile
1+0 records in
1+0 records out
4096 bytes (4.1 kB, 4.0 KiB) copied, 0.000306202 s, 13.4 MB/s
[root@z1 ~]# scp /etc/crypt_keyfile root@z2.example.com:/etc/
```

3. Distribuez le fichier clé cryptographique aux autres nœuds de la grappe, en utilisant le paramètre **-p** pour préserver les autorisations que vous avez définies.

```
[root@z1 ~]# scp -p /etc/crypt_keyfile root@z2.example.com:/etc/
```

4. Créez le périphérique crypté sur le volume LVM où vous configurerez le système de fichiers GFS2 crypté.

```
[root@z1 ~]# cryptsetup luksFormat /dev/shared_vg1/shared_lv1 --type luks2 --key-
file=/etc/crypt_keyfile
WARNING!
========
This will overwrite data on /dev/shared_vg1/shared_lv1 irrevocably.

Are you sure? (Type 'yes' in capital letters): YES
```

5. Créez la ressource cryptographique dans le cadre du groupe de volumes **shared_vg1**.

```
[root@z1 ~]# pcs resource create crypt --group shared_vg1 ocf:heartbeat:crypt
crypt_dev="luks_lv1" crypt_type=luks2 key_file=/etc/crypt_keyfile
encrypted_dev="/dev/shared_vg1/shared_lv1"
```

Verification steps

Assurez-vous que la ressource crypt a créé la clé de cryptage, qui dans cet exemple est **/dev/mapper/luks_lv1**.

```
[root@z1 ~]# ls -l /dev/mapper/
...
lrwxrwxrwx 1 root root 7 Mar 4 09:52 luks_lv1 -> ../dm-3
...
```

8.2.3. Formatez le volume logique crypté avec un système de fichiers GFS2 et créez une ressource de système de fichiers pour le cluster

Conditions préalables

- Vous avez chiffré le volume logique et créé une ressource cryptée.

Procédure

1. Sur un nœud du cluster, formatez le volume avec un système de fichiers GFS2. Un journal est nécessaire pour chaque nœud qui monte le système de fichiers. Veillez à créer suffisamment de journaux pour chacun des nœuds de votre grappe. Le format du nom de la table de verrouillage est *ClusterName:FSName*, où *ClusterName* est le nom de la grappe pour laquelle le système de fichiers GFS2 est créé et *FSName* est le nom du système de fichiers, qui doit être unique pour tous les systèmes de fichiers **lock_dlm** de la grappe.

```
[root@z1 ~]# mkfs.gfs2 -j3 -p lock_dlm -t my_cluster:gfs2-demo1 /dev/mapper/luks_lv1
/dev/mapper/luks_lv1 is a symbolic link to /dev/dm-3
This will destroy any data on /dev/dm-3
Are you sure you want to proceed? [y/n] y
Discarding device contents (may take a while on large devices): Done
Adding journals: Done
Building resource groups: Done
Creating quota file: Done
Writing superblock and syncing: Done
Device:                    /dev/mapper/luks_lv1
Block size:                4096
Device size:               4.98 GB (1306624 blocks)
Filesystem size:           4.98 GB (1306622 blocks)
Journals:                  3
Journal size:              16MB
Resource groups:           23
Locking protocol:          "lock_dlm"
Lock table:                "my_cluster:gfs2-demo1"
UUID:                      de263f7b-0f12-4d02-bbb2-56642fade293
```

2. Créez une ressource de système de fichiers pour monter automatiquement le système de fichiers GFS2 sur tous les nœuds.
   N'ajoutez pas le système de fichiers au fichier **/etc/fstab** car il sera géré comme une ressource de cluster Pacemaker. Les options de montage peuvent être spécifiées dans le cadre de la configuration de la ressource à l'aide de la commande **options=*options***. Exécutez la commande **pcs resource describe Filesystem** pour obtenir toutes les options de configuration.

   La commande suivante crée la ressource du système de fichiers. Cette commande ajoute la ressource au groupe de ressources qui comprend la ressource de volume logique pour ce système de fichiers.

```
[root@z1 ~]# pcs resource create sharedfs1 --group shared_vg1
ocf:heartbeat:Filesystem device="/dev/mapper/luks_lv1" directory="/mnt/gfs1"
fstype="gfs2" options=noatime op monitor interval=10s on-fail=fence
```

Verification steps

1. Vérifiez que le système de fichiers GFS2 est monté sur les deux nœuds du cluster.

```
[root@z1 ~]# mount | grep gfs2
/dev/mapper/luks_lv1 on /mnt/gfs1 type gfs2 (rw,noatime,seclabel)

[root@z2 ~]# mount | grep gfs2
/dev/mapper/luks_lv1 on /mnt/gfs1 type gfs2 (rw,noatime,seclabel)
```

2. Vérifier l'état de la grappe.

```
[root@z1 ~]# pcs status --full
Cluster name: my_cluster
[...]

Full list of resources:

  smoke-apc      (stonith:fence_apc):    Started z1.example.com
```

```
  Clone Set: locking-clone [locking]
      Resource Group: locking:0
          dlm    (ocf::pacemaker:controld):      Started z2.example.com
          lvmlockd       (ocf::heartbeat:lvmlockd):      Started z2.example.com
      Resource Group: locking:1
          dlm    (ocf::pacemaker:controld):      Started z1.example.com
          lvmlockd       (ocf::heartbeat:lvmlockd):      Started z1.example.com
    Started: [ z1.example.com z2.example.com ]
  Clone Set: shared_vg1-clone [shared_vg1]
    Resource Group: shared_vg1:0
          sharedlv1      (ocf::heartbeat:LVM-activate):  Started z2.example.com
          crypt       (ocf::heartbeat:crypt) Started z2.example.com
          sharedfs1      (ocf::heartbeat:Filesystem):    Started z2.example.com
   Resource Group: shared_vg1:1
          sharedlv1      (ocf::heartbeat:LVM-activate):  Started z1.example.com
          crypt      (ocf::heartbeat:crypt)  Started z1.example.com
          sharedfs1      (ocf::heartbeat:Filesystem):    Started z1.example.com
        Started:  [z1.example.com z2.example.com ]
...
```

Ressources supplémentaires

- Configuration des systèmes de fichiers GFS2

CHAPITRE 9. LES TRACEPOINTS DE GFS2 ET L'INTERFACE DEBUGFS DE GLOCK

Cette documentation sur les tracepoints de GFS2 et l'interface glock **debugfs** est destinée aux utilisateurs avancés qui sont familiers avec les systèmes de fichiers internes et qui souhaitent en savoir plus sur la conception de GFS2 et sur la manière de déboguer les problèmes spécifiques à GFS2.

Les sections suivantes décrivent les tracepoints GFS2 et le fichier GFS2 **glocks**.

9.1. TYPES DE POINTS D'APPUI GFS2

Il existe actuellement trois types de tracepoints GFS2 : *glock* (prononcé "gee-lock") tracepoints, *bmap* tracepoints et *log* tracepoints. Ils peuvent être utilisés pour surveiller un système de fichiers GFS2 en cours d'exécution et fournir des informations supplémentaires à celles qui peuvent être obtenues avec les options de débogage prises en charge dans les versions précédentes de Red Hat Enterprise Linux. Les points de contrôle sont particulièrement utiles lorsqu'un problème, tel qu'un blocage ou un problème de performance, est reproductible et que la sortie du point de contrôle peut être obtenue pendant l'opération problématique. Dans GFS2, les glocks sont le principal mécanisme de contrôle du cache et ils sont la clé pour comprendre les performances du cœur de GFS2. Les points de contrôle bmap (block map) peuvent être utilisés pour surveiller les allocations de blocs et le mappage de blocs (recherche de blocs déjà alloués dans l'arborescence des métadonnées du disque) au fur et à mesure qu'ils se produisent et pour vérifier tout problème lié à la localité de l'accès. Les tracepoints de journal suivent les données écrites et libérées du journal et peuvent fournir des informations utiles sur cette partie de GFS2.

Les tracepoints sont conçus pour être aussi génériques que possible, ce qui devrait signifier qu'il ne sera pas nécessaire de modifier l'API au cours de Red Hat Enterprise Linux 8. D'un autre côté, les utilisateurs de cette interface doivent savoir qu'il s'agit d'une interface de débogage et qu'elle ne fait pas partie de l'ensemble des API normales de Red Hat Enterprise Linux 8, et qu'en tant que telle, Red Hat ne garantit pas que des changements ne seront pas apportés à l'interface des tracepoints de GFS2.

Les tracepoints sont une caractéristique générique de Red Hat Enterprise Linux et leur champ d'application va bien au-delà de GFS2. Ils sont notamment utilisés pour mettre en œuvre l'infrastructure **blktrace** et les tracepoints **blktrace** peuvent être utilisés en combinaison avec ceux de GFS2 pour obtenir une image plus complète des performances du système. En raison du niveau auquel les tracepoints fonctionnent, ils peuvent produire d'importants volumes de données en très peu de temps. Ils sont conçus pour exercer une charge minimale sur le système lorsqu'ils sont activés, mais il est inévitable qu'ils aient un certain effet. Le filtrage des événements par divers moyens peut aider à réduire le volume de données et à se concentrer sur l'obtention des informations utiles à la compréhension d'une situation particulière.

9.2. TRACEPOINTS

Les points de traçage se trouvent dans le répertoire **/sys/kernel/debug/tracing/**, à condition que **debugfs** soit monté à l'emplacement standard du répertoire **/sys/kernel/debug**. Le sous-répertoire **events** contient tous les événements de traçage qui peuvent être spécifiés et, à condition que le module **gfs2** soit chargé, il y aura un sous-répertoire **gfs2** contenant d'autres sous-répertoires, un pour chaque événement GFS2. Le contenu du répertoire **/sys/kernel/debug/tracing/events/gfs2** devrait ressembler à peu près à ce qui suit :

```
[root@chywoon gfs2]# ls
enable          gfs2_bmap       gfs2_glock_queue        gfs2_log_flush
filter          gfs2_demote_rq  gfs2_glock_state_change  gfs2_pin
gfs2_block_alloc  gfs2_glock_put  gfs2_log_blocks         gfs2_promote
```

Pour activer tous les tracepoints GFS2, entrez la commande suivante :

```
[root@chywoon gfs2]# echo -n 1 >/sys/kernel/debug/tracing/events/gfs2/enable
```

Pour activer un point de contrôle spécifique, il existe un fichier **enable** dans chacun des sous-répertoires d'événements individuels. Il en va de même pour le fichier **filter** qui peut être utilisé pour définir un filtre d'événement pour chaque événement ou ensemble d'événements. La signification des différents événements est expliquée plus en détail ci-dessous.

La sortie des tracepoints est disponible en format ASCII ou binaire. Cette annexe ne couvre pas actuellement l'interface binaire. L'interface ASCII est disponible de deux manières. Pour obtenir la liste du contenu actuel du tampon circulaire, vous pouvez entrer la commande suivante :

```
[root@chywoon gfs2]# cat /sys/kernel/debug/tracing/trace
```

Cette interface est utile lorsque vous utilisez un processus de longue durée pendant un certain temps et que, après un événement, vous souhaitez consulter les dernières informations capturées dans la mémoire tampon. Une autre interface, **/sys/kernel/debug/tracing/trace_pipe**, peut être utilisée lorsque toutes les données de sortie sont nécessaires. Les événements sont lus dans ce fichier au fur et à mesure qu'ils se produisent ; aucune information historique n'est disponible via cette interface. Le format de sortie est le même pour les deux interfaces et est décrit pour chacun des événements GFS2 dans les sections suivantes de cette annexe.

Un utilitaire appelé **trace-cmd** est disponible pour lire les données des points de contrôle. Pour plus d'informations sur cet utilitaire, voir http://lwn.net/Articles/341902/. L'utilitaire **trace-cmd** peut être utilisé de la même manière que l'utilitaire **strace**, par exemple pour exécuter une commande tout en recueillant des données de traçage à partir de diverses sources.

9.3. GLOCKS

Pour comprendre GFS2, le concept le plus important, et celui qui le différencie des autres systèmes de fichiers, est celui des glocks. Dans le code source, un glock est une structure de données qui réunit le DLM et la mise en cache en une seule machine à états. Chaque glock a une relation 1:1 avec un seul verrou DLM et fournit un cache pour cet état de verrouillage afin que les opérations répétitives effectuées à partir d'un seul nœud du système de fichiers n'aient pas à appeler le DLM à plusieurs reprises, ce qui permet d'éviter un trafic réseau inutile. Il existe deux grandes catégories de glocks, ceux qui mettent en cache les métadonnées et ceux qui ne le font pas. Les glocks d'inodes et les glocks de groupes de ressources mettent tous deux des métadonnées en cache, tandis que les autres types de glocks ne mettent pas de métadonnées en cache. Le glock inode est également impliqué dans la mise en cache de données en plus des métadonnées et possède la logique la plus complexe de tous les glocks.

Tableau 9.1. Modes Glock et modes de verrouillage DLM

Mode Glock	Mode de verrouillage DLM	Notes
ONU	IV/NL	Déverrouillé (pas de serrure DLM associée au glock ou serrure NL en fonction de l'indicateur I)
SH	PR	Verrouillage partagé (lecture protégée)

Mode Glock	Mode de verrouillage DLM	Notes
EX	EX	Serrure exclusive
DF	CW	Différé (écriture simultanée) utilisé pour Direct I/O et le gel du système de fichiers

Les glocks restent en mémoire jusqu'à ce qu'ils soient déverrouillés (à la demande d'un autre nœud ou de la VM) et qu'il n'y ait plus d'utilisateurs locaux. Ils sont alors retirés de la table de hachage des glocks et libérés. Lorsqu'un glock est créé, le verrou DLM n'est pas immédiatement associé au glock. Le verrou DLM est associé au cadenas lors de la première demande au DLM, et si cette demande aboutit, le drapeau "I" (initial) est activé sur le cadenas. Le tableau "Glock Flags" dans The glock debugfs interface montre la signification des différents drapeaux de glock. Une fois que le DLM a été associé au glock, le verrou DLM restera toujours au moins en mode NL (Null) jusqu'à ce que le glock soit libéré. La rétrogradation du verrou DLM de NL à déverrouillé est toujours la dernière opération dans la vie d'un cadenas.

Chaque glock peut être associé à un certain nombre de "détenteurs", chacun d'entre eux représentant une demande de verrouillage de la part des couches supérieures. Les appels système relatifs à GFS2 mettent en file d'attente et retirent de la file d'attente les détenteurs du glock afin de protéger la section critique du code.

La machine à états de glock est basée sur une file d'attente. Pour des raisons de performance, les tasklets seraient préférables ; cependant, dans l'implémentation actuelle, nous devons soumettre des E/S à partir de ce contexte, ce qui interdit leur utilisation.

NOTE

Les files d'attente ont leurs propres tracepoints qui peuvent être utilisés en combinaison avec les tracepoints de GFS2.

Le tableau suivant montre quel état peut être mis en cache dans chacun des modes de verrouillage et si cet état mis en cache peut être sale. Cela s'applique aux verrous d'inodes et de groupes de ressources, bien qu'il n'y ait pas de composant de données pour les verrous de groupes de ressources, seulement des métadonnées.

Tableau 9.2. Modes et types de données du Glock

Mode Glock	Données du cache	Métadonnées du cache	Données sales	Métadonnées sales
ONU	Non	Non	Non	Non
SH	Oui	Oui	Non	Non
DF	Non	Oui	Non	Non
EX	Oui	Oui	Oui	Oui

9.4. L'INTERFACE DEBUGFS DE GLOCK

L'interface glock **debugfs** permet de visualiser l'état interne des glocks et des détenteurs et comprend également des détails sommaires sur les objets verrouillés dans certains cas. Chaque ligne du fichier commence soit par G : sans indentation (ce qui fait référence au glock lui-même), soit par une lettre différente, indentée d'un seul espace, et faisant référence aux structures associées au glock immédiatement au-dessus d'elle dans le fichier (H : est un support, I : un inode, et R : un groupe de ressources). Voici un exemple de ce que pourrait être le contenu de ce fichier :

```
G:  s:SH n:5/75320 f:I t:SH d:EX/0 a:0 r:3
 H: s:SH f:EH e:0 p:4466 [postmark] gfs2_inode_lookup+0x14e/0x260 [gfs2]
G:  s:EX n:3/258028 f:yI t:EX d:EX/0 a:3 r:4
 H: s:EX f:tH e:0 p:4466 [postmark] gfs2_inplace_reserve_i+0x177/0x780 [gfs2]
 R: n:258028 f:05 b:22256/22256 i:16800
G:  s:EX n:2/219916 f:yfI t:EX d:EX/0 a:0 r:3
 I: n:75661/219916 t:8 f:0x10 d:0x00000000 s:7522/7522
G:  s:SH n:5/127205 f:I t:SH d:EX/0 a:0 r:3
 H: s:SH f:EH e:0 p:4466 [postmark] gfs2_inode_lookup+0x14e/0x260 [gfs2]
G:  s:EX n:2/50382 f:yfI t:EX d:EX/0 a:0 r:2
G:  s:SH n:5/302519 f:I t:SH d:EX/0 a:0 r:3
 H: s:SH f:EH e:0 p:4466 [postmark] gfs2_inode_lookup+0x14e/0x260 [gfs2]
G:  s:SH n:5/313874 f:I t:SH d:EX/0 a:0 r:3
 H: s:SH f:EH e:0 p:4466 [postmark] gfs2_inode_lookup+0x14e/0x260 [gfs2]
G:  s:SH n:5/271916 f:I t:SH d:EX/0 a:0 r:3
 H: s:SH f:EH e:0 p:4466 [postmark] gfs2_inode_lookup+0x14e/0x260 [gfs2]
G:  s:SH n:5/312732 f:I t:SH d:EX/0 a:0 r:3
 H: s:SH f:EH e:0 p:4466 [postmark] gfs2_inode_lookup+0x14e/0x260 [gfs2]
```

L'exemple ci-dessus est une série d'extraits (d'un fichier d'environ 18 Mo) générés par la commande **cat /sys/kernel/debug/gfs2/unity:myfs/glocks >my.lock** lors de l'exécution du test de référence postmark sur un système de fichiers GFS2 à nœud unique. Les glocks de la figure ont été sélectionnés afin de montrer certaines des caractéristiques les plus intéressantes des dumps de glocks.

Les états de verrouillage sont EX (exclusif), DF (différé), SH (partagé) ou UN (déverrouillé). Ces états correspondent directement aux modes de verrouillage DLM, à l'exception de UN qui peut représenter soit l'état de verrouillage DLM nul, soit le fait que GFS2 ne détient pas de verrouillage DLM (en fonction de l'indicateur I, comme expliqué ci-dessus). Le champ s : du glock indique l'état actuel du verrou et le même champ dans le holder indique le mode demandé. Si le verrou est accordé, le détenteur aura le bit H activé dans ses drapeaux (champ f :). Dans le cas contraire, le bit d'attente W est activé.

Le champ n : (numéro) indique le numéro associé à chaque élément. Pour les glocks, il s'agit du numéro de type suivi du numéro de glock. Ainsi, dans l'exemple ci-dessus, le premier glock est n:5/75320, ce qui indique un glock **iopen** lié à l'inode 75320. Dans le cas des glocks d'inodes et de **iopen**, le numéro de glock est toujours identique au numéro de bloc de disque de l'inode.

NOTE

Les numéros de glock (champ n :) dans le fichier debugfs glocks sont en hexadécimal, alors que la sortie tracepoints les indique en décimal. C'est pour des raisons historiques ; les numéros de glock ont toujours été écrits en hexadécimal, mais la décimale a été choisie pour les tracepoints afin que les numéros puissent être facilement comparés avec les autres sorties de tracepoints (de **blktrace** par exemple) et avec la sortie de **stat**(1).

La liste complète de tous les drapeaux, tant pour le détenteur que pour le glock, figure dans le tableau "Glock Flags", ci-dessous, et dans le tableau "Glock Holder Flags", dans la rubrique "Glock holders" (

détenteurs de glock) . Le contenu des blocs de valeurs de verrouillage n'est actuellement pas disponible via l'interface glock **debugfs**.

Le tableau suivant indique la signification des différents types de glock.

Tableau 9.3. Types de Glock

Numéro de type	Type de serrure	Utilisation
1	trans	Verrouillage des transactions
2	inode	Métadonnées et données des inodes
3	rgrp	Métadonnées du groupe de ressources
4	méta	Le superbloc
5	iopen	Dernière détection rapprochée de l'inode
6	troupeau	**flock**(2) syscall
8	quota	Opérations de quotas
9	journal	Journal mutex

L'un des drapeaux les plus importants de glock est le drapeau l (locked). Il s'agit du bit de verrouillage utilisé pour arbitrer l'accès à l'état glock lorsqu'un changement d'état doit être effectué. Il est activé lorsque la machine d'état est sur le point d'envoyer une demande de verrouillage à distance via le DLM, et n'est désactivé que lorsque l'opération complète a été effectuée. Parfois, cela peut signifier que plus d'une demande de verrouillage a été envoyée, avec diverses invalidations se produisant entre-temps.

Le tableau suivant indique la signification des différents drapeaux du glock.

Tableau 9.4. Drapeaux Glock

Drapeau	Nom	Signification
d	Rétrogradation en cours	Une demande de rétrogradation différée (à distance)
D	Rétrograder	Une demande de rétrogradation (locale ou à distance)
f	Rinçage des grumes	Le journal doit être engagé avant de sortir ce glock

Drapeau	Nom	Signification
F	Congelé	Les réponses des nœuds distants sont ignorées - la récupération est en cours.
i	Invalidation en cours	En cours d'invalidation des pages sous ce glock
I	Initiale	Fixé lorsque la serrure DLM est associée à ce glock
l	Verrouillé	Le glock est en train de changer d'état
L	LRU	Défini lorsque le glock est sur la liste LRU`
o	Objet	Défini lorsque le glock est associé à un objet (c'est-à-dire un inode pour les glocks de type 2 et un groupe de ressources pour les glocks de type 3)
p	Rétrogradation en cours	Le glock est en train de répondre à une demande de rétrogradation
q	En attente	Défini lorsqu'un détenteur est mis en file d'attente pour un glock, et effacé lorsque le glock est tenu, mais qu'il n'y a plus de détenteurs restants. Utilisé dans le cadre de l'algorithme qui calcule le temps de maintien minimum d'un glock.
r	Réponse en attente	La réponse reçue du nœud distant est en attente de traitement
y	Sale	Les données doivent être transférées sur le disque avant la mise en circulation de ce glock

Lorsqu'un rappel à distance est reçu d'un nœud qui souhaite obtenir un verrou dans un mode qui entre en conflit avec celui détenu par le nœud local, l'un ou l'autre des deux drapeaux D (demote) ou d (demote pending) est activé. Afin d'éviter les situations de famine lorsqu'il y a de la contention sur un verrou particulier, chaque verrou se voit attribuer un temps de maintien minimum. Un nœud qui n'a pas encore eu le verrou pendant le temps de maintien minimum est autorisé à conserver ce verrou jusqu'à ce que l'intervalle de temps ait expiré.

Si l'intervalle de temps a expiré, l'indicateur D (demote) est activé et l'état requis est enregistré. Dans ce

cas, la prochaine fois qu'il n'y aura pas de verrou accordé dans la file d'attente des détenteurs, le verrou sera rétrogradé. Si l'intervalle de temps n'a pas expiré, l'indicateur d (demote pending) est activé à la place. Cela permet également de programmer la machine à états pour effacer d (demote pending) et définir D (demote) lorsque le temps d'attente minimum a expiré.

L'indicateur I (initial) est activé lorsque le verrou DLM a été attribué au glock. Cela se produit lorsque le glock est utilisé pour la première fois et l'indicateur I reste activé jusqu'à ce que le glock soit finalement libéré (c'est-à-dire que le verrou DLM soit déverrouillé).

9.5. PORTE-BLOCS GLOCK

Le tableau suivant indique la signification des différents drapeaux du porte-bloc.

Tableau 9.5. Drapeaux pour porte-blocs Glock

Drapeau	Nom	Signification
a	Asynchrone	N'attendez pas le résultat de l'enquête sur le glock (le résultat de l'enquête sera communiqué plus tard)
A	Tous	Tout mode de verrouillage compatible est acceptable
c	Pas de cache	En cas de déverrouillage, rétrograder immédiatement la serrure DLM
e	Pas d'expiration	Ignorer les demandes ultérieures d'annulation de verrou
E	Exactement	Doit disposer d'un mode de verrouillage exact
F	Première	Fixé lorsque le détenteur est le premier à bénéficier de cette serrure
H	Titulaire	Indique que le verrou demandé est accordé
p	Priorité	Enqueue holder at the head of the queue
t	Essayer	Serrure "try" A \N- "try" - "try" - "try" - "try" - "try" - "try
T	Essayer 1CB	Un verrou "try" qui envoie un rappel

Drapeau	Nom	Signification
W	Attendre	Fixé pendant l'attente de l'achèvement de la demande

Les indicateurs de détenteurs les plus importants sont H (holder) et W (wait), comme indiqué précédemment, puisqu'ils sont activés respectivement sur les demandes de verrou accordées et sur les demandes de verrou en file d'attente. L'ordre des détenteurs dans la liste est important. S'il y a des détenteurs accordés, ils seront toujours en tête de la file d'attente, suivis par les détenteurs en file d'attente.

S'il n'y a pas de titulaire accordé, le premier titulaire de la liste sera celui qui déclenchera le prochain changement d'état. Les demandes de rétrogradation étant toujours considérées comme plus prioritaires que les demandes émanant du système de fichiers, il se peut qu'elles n'entraînent pas directement une modification de l'état demandé.

Le sous-système glock prend en charge deux types de verrous "try". Ceux-ci sont utiles à la fois parce qu'ils permettent de prendre des verrous hors de l'ordre normal (avec un back-off et un retry appropriés) et parce qu'ils peuvent être utilisés pour éviter des ressources utilisées par d'autres nœuds. Le verrou normal t (try) est exactement ce que son nom indique ; c'est un verrou "try" qui ne fait rien de spécial. Le verrou T (**try 1CB**), quant à lui, est identique au verrou t, à ceci près que le DLM envoie un rappel unique aux détenteurs de verrous incompatibles actuels. Le verrou T (**try 1CB**) est notamment utilisé avec les verrous **iopen**, qui servent à arbitrer entre les nœuds lorsque le compte **i_nlink** d'un inode est nul, et à déterminer lequel des nœuds sera responsable de la désallocation de l'inode. Le verrou **iopen** est normalement maintenu dans l'état partagé, mais lorsque le compte **i_nlink** devient nul et que →**evict_inode**() est appelé, il demande un verrou exclusif avec T (**try 1CB**). Il continuera à désallouer l'inode si le verrou est accordé. Si le verrou n'est pas accordé, le ou les nœuds qui empêchaient l'octroi du verrou marqueront leur(s) glock(s) du drapeau D (demote), qui est vérifié au moment de →**drop_inode**() afin de s'assurer que la désallocation n'est pas oubliée.

Cela signifie que les inodes dont le nombre de liens est nul mais qui sont encore ouverts seront désalloués par le nœud sur lequel se produit le dernier **close**(). De plus, au moment où le nombre de liens de l'inode est décrémenté à zéro, l'inode est marqué comme étant dans l'état spécial d'avoir un nombre de liens nul mais toujours en cours d'utilisation dans la carte bitmap du groupe de ressources. Cela fonctionne comme la liste des orphelins du système de fichiers ext3, en ce sens que cela permet à tout lecteur ultérieur de la carte bitmap de savoir qu'il y a potentiellement de l'espace qui pourrait être récupéré, et d'essayer de le récupérer.

9.6. GLOCK TRACEPOINTS

Les tracepoints sont également conçus pour pouvoir confirmer l'exactitude du contrôle du cache en les combinant avec la sortie de **blktrace** et avec la connaissance de la disposition sur le disque. Il est alors possible de vérifier qu'une E/S donnée a été émise et achevée sous le bon verrou, et qu'il n'y a pas de course.

Le tracepoint **gfs2_glock_state_change** est le plus important à comprendre. Il suit chaque changement d'état du glock depuis sa création initiale jusqu'à la rétrogradation finale qui se termine par **gfs2_glock_put** et la transition finale de NL à unlocked (déverrouillé). Le drapeau l (locked) glock est toujours activé avant qu'un changement d'état ne se produise et ne sera effacé qu'une fois celui-ci terminé. Lors d'un changement d'état, il n'y a jamais de détenteurs accordés (drapeau de détenteur de

cadenas H). S'il y a des titulaires en file d'attente, ils seront toujours dans l'état W (en attente). Lorsque le changement d'état est terminé, les titulaires peuvent être accordés, ce qui constitue la dernière opération avant l'effacement de l'indicateur l glock.

Le tracepoint **gfs2_demote_rq** garde la trace des demandes de rétrogradation, qu'elles soient locales ou distantes. En supposant qu'il y ait suffisamment de mémoire sur le nœud, les demandes de rétrogradation locales seront rarement vues, et le plus souvent elles seront créées par **umount** ou par une récupération occasionnelle de la mémoire. Le nombre de demandes de rétrogradation à distance est une mesure de la concurrence entre les nœuds pour un inode ou un groupe de ressources particulier.

Le point de contrôle **gfs2_glock_lock_time** fournit des informations sur la durée des requêtes adressées au DLM. L'indicateur de blocage (**b**) a été introduit dans le glock spécifiquement pour être utilisé en combinaison avec ce point de contrôle.

Lorsqu'un titulaire se voit accorder un verrou, **gfs2_promote** est appelé, ce qui se produit lors des étapes finales d'un changement d'état ou lorsqu'un verrou est demandé et qu'il peut être accordé immédiatement parce que l'état glock a déjà mis en cache un verrou d'un mode approprié. Si le détenteur est le premier à se voir accorder ce verrou, l'indicateur f (first) est activé sur ce détenteur. Cette fonction n'est actuellement utilisée que par les groupes de ressources.

9.7. CARTE DES POINTS DE REPÈRE

Le mappage des blocs est une tâche essentielle pour tout système de fichiers. GFS2 utilise un système traditionnel basé sur la cartographie avec deux bits par bloc. L'objectif principal des tracepoints dans ce sous-système est de permettre le contrôle du temps nécessaire à l'allocation et au mappage des blocs.

Le tracepoint **gfs2_bmap** est appelé deux fois pour chaque opération bmap : une fois au début pour afficher la requête bmap, et une fois à la fin pour afficher le résultat. Il est ainsi facile de faire correspondre les demandes et les résultats et de mesurer le temps nécessaire pour mapper des blocs dans différentes parties du système de fichiers, différents décalages de fichiers ou même différents fichiers. Il est également possible de voir quelles sont les tailles d'étendue moyennes renvoyées par rapport à celles demandées.

Le point de contrôle **gfs2_rs** suit les réservations de blocs au fur et à mesure de leur création, de leur utilisation et de leur destruction dans l'allocateur de blocs.

Pour garder une trace des blocs alloués, **gfs2_block_alloc** est appelé non seulement lors des allocations, mais aussi lors de la libération des blocs. Comme les allocations sont toutes référencées en fonction de l'inode auquel le bloc est destiné, il est possible de savoir quels blocs physiques appartiennent à quels fichiers dans un système de fichiers actif. Ceci est particulièrement utile lorsqu'il est combiné avec **blktrace**, qui montrera les schémas d'E/S problématiques qui peuvent alors être renvoyés aux inodes pertinents en utilisant la cartographie obtenue au moyen de ce tracepoint.

Direct I/O (**iomap**) est une politique de cache alternative qui permet aux transferts de données de fichiers de se produire directement entre le disque et la mémoire tampon de l'utilisateur. Cela présente des avantages dans les situations où le taux d'atteinte du cache est censé être faible. Les points de trace **gfs2_iomap_start** et **gfs2_iomap_end** retracent ces opérations et peuvent être utilisés pour garder une trace du mappage utilisant l'E/S directe, les positions sur le système de fichiers de l'E/S directe ainsi que le type d'opération.

9.8. LOG TRACEPOINTS

Les tracepoints de ce sous-système suivent les blocs ajoutés et retirés du journal (**gfs2_pin**), ainsi que le temps nécessaire à la validation des transactions dans le journal (**gfs2_log_flush**). Cela peut s'avérer très utile pour déboguer les problèmes de performance de la journalisation.

Le point de contrôle **gfs2_log_blocks** garde la trace des blocs réservés dans le journal, ce qui peut aider à montrer si le journal est trop petit pour la charge de travail, par exemple.

Le point de contrôle **gfs2_ail_flush** est similaire au point de contrôle **gfs2_log_flush** dans la mesure où il suit le début et la fin des vidanges de la liste AIL. La liste AIL contient des tampons qui sont passés par le journal, mais qui n'ont pas encore été réécrits à leur place. Elle est périodiquement vidée afin de libérer de l'espace pour le système de fichiers, ou lorsqu'un processus demande un **sync** ou un **fsync**.

9.9. STATISTIQUES GLOCK

GFS2 maintient des statistiques qui peuvent aider à suivre ce qui se passe dans le système de fichiers. Cela vous permet de repérer les problèmes de performance.

GFS2 gère deux compteurs :

- **dcount**qui compte le nombre d'opérations DLM demandées. Cela montre combien de données ont été utilisées pour les calculs de moyenne/variance.
- **qcount**qui compte le nombre d'opérations de niveau **syscall** demandées. En général, **qcount** est égal ou supérieur à **dcount**.

En outre, GFS2 conserve trois paires moyenne/variance. Les paires moyenne/variance sont des estimations exponentielles lissées et l'algorithme utilisé est celui qui sert à calculer les temps d'aller-retour dans le code du réseau.

Les paires de moyenne et de variance conservées dans GFS2 ne sont pas mises à l'échelle, mais sont exprimées en nanosecondes entières.

- srtt/srttvar : Temps d'aller-retour lissé pour les opérations non bloquantes
- srttb/srttvarb : Temps d'aller-retour lissé pour les opérations de blocage
- irtt/irttvar : Temps inter-requêtes (par exemple, temps entre les requêtes DLM)

Une requête non bloquante est une requête qui se termine immédiatement, quel que soit l'état du verrou DLM en question. Cela signifie actuellement toute demande lorsque (a) l'état actuel du verrou est exclusif (b) l'état demandé est soit nul soit déverrouillé ou (c) l'indicateur "try lock" est activé. Une demande bloquante couvre toutes les autres demandes de verrouillage.

Les temps les plus longs sont meilleurs pour les IRTT, tandis que les temps les plus courts sont meilleurs pour les RTT.

Les statistiques sont conservées dans deux fichiers **sysfs**:

- Le fichier **glstats**. Ce fichier est similaire au fichier **glocks**, sauf qu'il contient des statistiques, avec un glock par ligne. Les données sont initialisées à partir des données "per cpu" du type de glock pour lequel le glock est créé (à l'exception des compteurs, qui sont mis à zéro). Ce fichier peut être très volumineux.
- Le fichier **lkstats**. Il contient des statistiques "par processeur" pour chaque type de glock. Il contient une statistique par ligne, dans laquelle chaque colonne représente un cœur de processeur. Il y a huit lignes par type de glock, les types se succédant les uns aux autres.

9.10. RÉFÉRENCES

Pour plus d'informations sur les tracepoints et le fichier GFS2 **glocks**, voir les ressources suivantes :

- Pour plus d'informations sur les règles de verrouillage interne des glocks, voir https://git.kernel.org/pub/scm/linux/kernel/git/torvalds/linux.git/tree/Documentation/filesystem glocks.rst.

- Pour plus d'informations sur la traçabilité des événements, voir https://git.kernel.org/pub/scm/linux/kernel/git/torvalds/linux.git/tree/Documentation/trace/eve

- Pour plus d'informations sur l'utilitaire **trace-cmd**, voir http://lwn.net/Articles/341902/.

CHAPITRE 10. SURVEILLANCE ET ANALYSE DES SYSTÈMES DE FICHIERS GFS2 À L'AIDE DE PERFORMANCE CO-PILOT (PCP)

Performance Co-Pilot (PCP) peut aider à surveiller et à analyser les systèmes de fichiers GFS2. La surveillance des systèmes de fichiers GFS2 dans PCP est assurée par le module GFS2 PMDA de Red Hat Enterprise Linux, disponible via le paquetage **pcp-pmda-gfs2**.

Le PMDA GFS2 fournit un certain nombre de mesures données par les statistiques GFS2 fournies dans le sous-système **debugfs**. Lorsqu'il est installé, le PMDA expose les valeurs données dans les fichiers **glocks**, **glstats**, et **sbstats**. Ceux-ci rapportent des ensembles de statistiques sur chaque système de fichiers GFS2 monté. Le PMDA utilise également les tracepoints du noyau GFS2 exposés par le Kernel Function Tracer (**ftrace**).

10.1. INSTALLATION DU PMDA GFS2

Pour fonctionner correctement, le PMDA GFS2 nécessite que le système de fichiers **debugfs** soit monté. Si le système de fichiers **debugfs** n'est pas monté, exécutez les commandes suivantes avant d'installer le GFS2 PMDA :

```
# mkdir /sys/kernel/debug
# mount -t debugfs none /sys/kernel/debug
```

Le PMDA GFS2 n'est pas activé dans le cadre de l'installation par défaut. Pour utiliser la surveillance métrique de GFS2 via PCP, vous devez l'activer après l'installation.

Exécutez les commandes suivantes pour installer PCP et activer le PMDA GFS2. Notez que le script d'installation du PMDA doit être exécuté en tant que root.

```
# dnf install pcp pcp-pmda-gfs2
# cd /var/lib/pcp/pmdas/gfs2
# ./Install
Updating the Performance Metrics Name Space (PMNS) ...
Terminate PMDA if already installed ...
Updating the PMCD control file, and notifying PMCD ...
Check gfs2 metrics have appeared ... 346 metrics and 255 values
```

10.2. AFFICHAGE D'INFORMATIONS SUR LES MESURES DE PERFORMANCE DISPONIBLES AVEC L'OUTIL PMINFO

L'outil **pminfo** affiche des informations sur les mesures de performance disponibles. Les exemples suivants montrent différentes mesures GFS2 que vous pouvez afficher avec cet outil.

10.2.1. Examen du nombre de structures glock qui existent actuellement par système de fichiers

Les mesures GFS2 glock donnent un aperçu du nombre de structures glock actuellement intégrées pour chaque système de fichiers GFS2 monté et de leurs états de verrouillage. Dans GFS2, un glock est une structure de données qui rassemble le DLM et la mise en cache en une seule machine à états. Chaque glock a une correspondance 1:1 avec un seul verrou DLM et fournit un cache pour les états de verrouillage afin que les opérations répétitives effectuées sur un seul nœud n'aient pas à appeler le DLM à plusieurs reprises, réduisant ainsi le trafic réseau inutile.

La commande **pminfo** suivante affiche une liste du nombre de glocks par système de fichiers GFS2 monté en fonction de leur mode de verrouillage.

```
# pminfo -f gfs2.glocks

gfs2.glocks.total
    inst [0 or "afc_cluster:data"] value 43680
    inst [1 or "afc_cluster:bin"] value 2091

gfs2.glocks.shared
    inst [0 or "afc_cluster:data"] value 25
    inst [1 or "afc_cluster:bin"] value 25

gfs2.glocks.unlocked
    inst [0 or "afc_cluster:data"] value 43652
    inst [1 or "afc_cluster:bin"] value 2063

gfs2.glocks.deferred
    inst [0 or "afc_cluster:data"] value 0
    inst [1 or "afc_cluster:bin"] value 0

gfs2.glocks.exclusive
    inst [0 or "afc_cluster:data"] value 3
    inst [1 or "afc_cluster:bin"] value 3
```

10.2.2. Examen du nombre de structures glock existant par type de système de fichiers

Les métriques GFS2 glstats donnent le nombre de chaque type de bloc existant pour chaque système de fichiers. Un grand nombre d'entre eux seront normalement de type inode (inode et métadonnées) ou groupe de ressources (métadonnées de groupe de ressources).

La commande suivante **pminfo** affiche une liste du nombre de chaque type de Glock par système de fichiers GFS2 monté.

```
# pminfo -f gfs2.glstats

gfs2.glstats.total
    inst [0 or "afc_cluster:data"] value 43680
    inst [1 or "afc_cluster:bin"] value 2091

gfs2.glstats.trans
    inst [0 or "afc_cluster:data"] value 3
    inst [1 or "afc_cluster:bin"] value 3

gfs2.glstats.inode
    inst [0 or "afc_cluster:data"] value 17
    inst [1 or "afc_cluster:bin"] value 17

gfs2.glstats.rgrp
    inst [0 or "afc_cluster:data"] value 43642
    inst [1 or "afc_cluster:bin"] value 2053

gfs2.glstats.meta
    inst [0 or "afc_cluster:data"] value 1
```

```
    inst [1 or "afc_cluster:bin"] value 1

gfs2.glstats.iopen
    inst [0 or "afc_cluster:data"] value 16
    inst [1 or "afc_cluster:bin"] value 16

gfs2.glstats.flock
    inst [0 or "afc_cluster:data"] value 0
    inst [1 or "afc_cluster:bin"] value 0

gfs2.glstats.quota
    inst [0 or "afc_cluster:data"] value 0
    inst [1 or "afc_cluster:bin"] value 0

gfs2.glstats.journal
    inst [0 or "afc_cluster:data"] value 1
    inst [1 or "afc_cluster:bin"] value 1
```

10.2.3. Vérification du nombre de structures glock qui sont dans un état d'attente

Les indicateurs de détente les plus importants sont H (holder : indique que le verrou demandé est accordé) et W (wait : activé pendant l'attente de la fin de la demande). Ces drapeaux sont activés respectivement sur les demandes de verrou accordées et sur les demandes de verrou en file d'attente.

La commande suivante **pminfo** affiche une liste du nombre de glocks avec l'indicateur de détenteur Wait (W) pour chaque système de fichiers GFS2 monté.

```
# pminfo -f gfs2.holders.flags.wait

gfs2.holders.flags.wait
    inst [0 or "afc_cluster:data"] value 0
    inst [1 or "afc_cluster:bin"] value 0
```

Si vous voyez un certain nombre de demandes en attente sur un verrou de groupe de ressources, il peut y avoir plusieurs raisons à cela. L'une d'elles est qu'il y a un grand nombre de nœuds par rapport au nombre de groupes de ressources dans le système de fichiers. Une autre raison est que le système de fichiers est presque plein (ce qui nécessite, en moyenne, des recherches plus longues pour les blocs libres). Dans les deux cas, la situation peut être améliorée en ajoutant de l'espace de stockage et en utilisant la commande **gfs2_grow** pour étendre le système de fichiers.

10.2.4. Vérification de la latence des opérations du système de fichiers à l'aide des mesures basées sur les points de contrôle du noyau

Le PMDA GFS2 prend en charge la collecte de métriques à partir des tracepoints du noyau GFS2. Par défaut, la lecture de ces métriques est désactivée. L'activation de ces métriques active les tracepoints du noyau GFS2 lorsque les métriques sont collectées afin de remplir les valeurs des métriques. Cela peut avoir un léger effet sur le débit des performances lorsque ces métriques de points de contrôle du noyau sont activées.

PCP fournit l'outil **pmstore**, qui vous permet de modifier les paramètres PMDA en fonction des valeurs métriques. Les paramètres **gfs2.control.*** permettent de basculer les tracepoints du noyau GFS2. L'exemple suivant utilise la commande **pmstore** pour activer tous les points de contrôle du noyau GFS2.

```
# pmstore gfs2.control.tracepoints.all 1
gfs2.control.tracepoints.all old value=0 new value=1
```

Lorsque cette commande est exécutée, le PMDA active tous les points de contrôle GFS2 dans le système de fichiers **debugfs**. Le tableau "Complete Metric List" dans Complete listing of available metrics for GFS2 in PCP explique chacun des points de contrôle et leur utilisation. Une explication de l'effet de chaque point de contrôle et de ses options est également disponible via le commutateur help dans **pminfo**.

Les mesures de promotion de GFS2 comptabilisent le nombre de demandes de promotion sur le système de fichiers. Ces demandes sont séparées par le nombre de demandes qui se sont produites lors de la première tentative et par les "autres" qui sont accordées après leur demande de promotion initiale. Une baisse du nombre de promotions à la première tentative et une augmentation du nombre de promotions "autres" peuvent indiquer des problèmes de contention de fichiers.

Les mesures de demandes de rétrogradation de GFS2, comme les mesures de demandes de promotion, comptent le nombre de demandes de rétrogradation qui se produisent sur le système de fichiers. Toutefois, ces demandes sont également réparties entre les demandes provenant du nœud actuel et les demandes provenant d'autres nœuds du système. Un grand nombre de demandes de rétrogradation provenant de nœuds distants peut indiquer une compétition entre deux nœuds pour un groupe de ressources donné.

L'outil **pminfo** affiche des informations sur les mesures de performances disponibles. Cette procédure permet d'afficher une liste du nombre de blocs avec l'indicateur Wait (W) holder pour chaque système de fichiers GFS2 monté. La commande **pminfo** suivante affiche une liste du nombre de glocks avec l'indicateur Wait (W) holder pour chaque système de fichiers GFS2 monté.

```
# pminfo -f gfs2.latency.grant.all gfs2.latency.demote.all

gfs2.latency.grant.all
    inst [0 or "afc_cluster:data"] value 0
    inst [1 or "afc_cluster:bin"] value 0

gfs2.latency.demote.all
    inst [0 or "afc_cluster:data"] value 0
    inst [1 or "afc_cluster:bin"] value 0
```

Il est conseillé de déterminer les valeurs générales observées lorsque la charge de travail fonctionne sans problème afin de pouvoir remarquer les changements de performance lorsque ces valeurs diffèrent de leur plage normale.

Par exemple, vous pourriez remarquer un changement dans le nombre de demandes de promotion qui attendent de se terminer plutôt que de se terminer du premier coup, ce que la sortie de la commande suivante vous permettrait de déterminer.

```
# pminfo -f gfs2.latency.grant.all gfs2.latency.demote.all

gfs2.tracepoints.promote.other.null_lock
    inst [0 or "afc_cluster:data"] value 0
    inst [1 or "afc_cluster:bin"] value 0

gfs2.tracepoints.promote.other.concurrent_read
    inst [0 or "afc_cluster:data"] value 0
    inst [1 or "afc_cluster:bin"] value 0
```

```
gfs2.tracepoints.promote.other.concurrent_write
    inst [0 or "afc_cluster:data"] value 0
    inst [1 or "afc_cluster:bin"] value 0

gfs2.tracepoints.promote.other.protected_read
    inst [0 or "afc_cluster:data"] value 0
    inst [1 or "afc_cluster:bin"] value 0

gfs2.tracepoints.promote.other.protected_write
    inst [0 or "afc_cluster:data"] value 0
    inst [1 or "afc_cluster:bin"] value 0

gfs2.tracepoints.promote.other.exclusive
    inst [0 or "afc_cluster:data"] value 0
    inst [1 or "afc_cluster:bin"] value 0
```

La sortie de la commande suivante vous permettra de déterminer une forte augmentation des demandes de rétrogradation à distance (en particulier si elles proviennent d'autres nœuds du cluster).

```
# pminfo -f gfs2.tracepoints.demote_rq.requested

gfs2.tracepoints.demote_rq.requested.remote
    inst [0 or "afc_cluster:data"] value 0
    inst [1 or "afc_cluster:bin"] value 0

gfs2.tracepoints.demote_rq.requested.local
    inst [0 or "afc_cluster:data"] value 0
    inst [1 or "afc_cluster:bin"] value 0
```

La sortie de la commande suivante pourrait indiquer une augmentation inexpliquée des vidanges de journaux.

```
# pminfo -f gfs2.tracepoints.log_flush.total]

gfs2.tracepoints.log_flush.total
    inst [0 or "afc_cluster:data"] value 0
    inst [1 or "afc_cluster:bin"] value 0
```

10.3. LISTE COMPLÈTE DES MESURES DISPONIBLES POUR GFS2 DANS PCP

Le tableau suivant décrit la liste complète des mesures de performance fournies par le paquetage **pcp-pmda-gfs2** pour les systèmes de fichiers GFS2.

Tableau 10.1. Liste complète des mesures

Nom de la métrique	Description
gfs2.glocks.*	Métriques concernant les informations collectées à partir du fichier glock stats (**glocks**) qui compte le nombre de glocks dans chaque état qui existe actuellement pour chaque système de fichiers GFS2 actuellement monté sur le système.

Nom de la métrique	Description
gfs2.glocks.flags.*	Gamme de métriques comptant le nombre de glocks qui existent avec les drapeaux de glocks donnés
gfs2.holders.*	Métriques concernant les informations collectées à partir du fichier glock stats (**glocks**) qui compte le nombre de glocks avec des détenteurs dans chaque état de verrouillage qui existe actuellement pour chaque système de fichiers GFS2 actuellement monté sur le système.
gfs2.holders.flags.*	Gamme de métriques comptant le nombre de porte-objets glocks avec les drapeaux de porte-objets donnés
gfs2.sbstats.*	Mesures de temps concernant les informations collectées à partir du fichier de statistiques des superblocs (**sbstats**) pour chaque système de fichiers GFS2 actuellement monté sur le système.
gfs2.glstats.*	Métriques concernant les informations collectées à partir du fichier glock stats (**glstats**) qui compte le nombre de chaque type de glock existant actuellement pour chaque système de fichiers GFS2 actuellement monté sur le système.
gfs2.latency.grant.*	Une mesure dérivée utilisant les données des points de repère **gfs2_glock_queue** et **gfs2_glock_state_change** pour calculer un temps de latence moyen en microsecondes pour que les demandes de subvention glock soient traitées pour chaque système de fichiers monté. Cette mesure est utile pour détecter les ralentissements potentiels du système de fichiers lorsque la latence d'octroi augmente.
gfs2.latency.demote.*	Mesure dérivée utilisant les données des points de référence **gfs2_glock_state_change** et **gfs2_demote_rq** pour calculer le temps de latence moyen en microsecondes nécessaire à l'exécution des requêtes de démotorisation de glock pour chaque système de fichiers monté. Cette mesure est utile pour détecter les ralentissements potentiels du système de fichiers lorsque le temps de latence augmente.
gfs2.latency.queue.*	Mesure dérivée utilisant les données du point de contrôle **gfs2_glock_queue** pour calculer une latence moyenne en microsecondes pour les demandes de file d'attente glock pour chaque système de fichiers monté.
gfs2.worst_glock.*	Mesure dérivée utilisant les données du point de contrôle **gfs2_glock_lock_time** pour calculer la perception du "pire verrou actuel" pour chaque système de fichiers monté. Cette mesure est utile pour découvrir les conflits potentiels entre les verrous et le ralentissement du système de fichiers si le même verrou est suggéré plusieurs fois.

Nom de la métrique	Description
gfs2.tracepoints.*	Métriques concernant la sortie des tracepoints GFS2 **debugfs** pour chaque système de fichiers actuellement monté sur le système. Chaque sous-type de ces métriques (un pour chaque tracepoint GFS2) peut être contrôlé individuellement, qu'il soit activé ou désactivé, à l'aide des métriques de contrôle.
gfs2.control.*	Les métriques de configuration sont utilisées pour activer ou désactiver l'enregistrement des métriques dans le PMDA. Les métriques de contrôle sont activées à l'aide de l'outil **pmstore**.

10.4. EFFECTUER UNE CONFIGURATION PCP MINIMALE POUR COLLECTER DES DONNÉES SUR LE SYSTÈME DE FICHIERS

Cette procédure décrit les instructions à suivre pour installer une configuration PCP minimale afin de collecter des statistiques sur Red Hat Enterprise Linux. Cette configuration implique l'ajout du nombre minimum de paquets sur un système de production nécessaire pour collecter des données en vue d'une analyse ultérieure.

L'archive **tar.gz** de la sortie **pmlogger** qui en résulte peut être analysée à l'aide d'autres outils PCP et comparée à d'autres sources d'informations sur les performances.

Procédure

1. Installez les paquets PCP requis.

```
# dnf install pcp pcp-pmda-gfs2
```

2. Activer le module GFS2 pour le PCP.

```
# cd /var/lib/pcp/pmdas/gfs2
# ./Install
```

3. Démarrez les services **pmcd** et **pmlogger**.

```
# systemctl start pmcd.service
# systemctl start pmlogger.service
```

4. Effectuer des opérations sur le système de fichiers GFS2.

5. Arrêtez les services **pmcd** et **pmlogger**.

```
# systemctl stop pmcd.service
# systemctl stop pmlogger.service
```

6. Rassemblez les résultats et enregistrez-les dans un fichier **tar.gz** dont le nom est basé sur le nom de l'hôte et sur la date et l'heure actuelles.

```
# cd /var/log/pcp/pmlogger
# tar -czf $(hostname).$(date+%F-%Hh%M).pcp.tar.gz $(hostname)
```

10.5. RESSOURCES SUPPLÉMENTAIRES

- Les tracepoints de GFS2 et l'interface debugfs de glock .
- Suivi des performances avec Performance Co-Pilot
- Index des articles, solutions, tutoriels et livres blancs de Performance Co-Pilot (PCP)