# Red Hat Enterprise Linux 8

## Configuração dos sistemas de arquivo GFS2

Um guia para a configuração e gerenciamento dos sistemas de arquivo GFS2

Last Updated: 2021-03-08

# Red Hat Enterprise Linux 8 Configuração dos sistemas de arquivo GFS2

Um guia para a configuração e gerenciamento dos sistemas de arquivo GFS2

Enter your first name here. Enter your surname here.
Enter your organisation's name here. Enter your organisational division here.
Enter your email address here.

## Nota Legal



## Resumo

Este guia fornece informações sobre configuração e gerenciamento dos sistemas de arquivo GFS2 para o Red Hat Enterprise Linux 8.

# Índice

# TORNANDO O CÓDIGO ABERTO MAIS INCLUSIVO

A Red Hat tem o compromisso de substituir a linguagem problemática em nosso código, documentação e propriedades da web. Estamos começando com estes quatro termos: master, slave, blacklist e whitelist. Por causa da enormidade deste esforço, estas mudanças serão implementadas gradualmente ao longo de vários lançamentos futuros. Para mais detalhes, veja a mensagem de nosso CTO Chris Wright.

# FORNECENDO FEEDBACK SOBRE A DOCUMENTAÇÃO DA RED HAT

Agradecemos sua contribuição em nossa documentação. Por favor, diga-nos como podemos melhorá-la. Para fazer isso:

- Para comentários simples sobre passagens específicas:
  1. Certifique-se de que você está visualizando a documentação no formato *Multi-page HTML*. Além disso, certifique-se de ver o botão **Feedback** no canto superior direito do documento.
  2. Use o cursor do mouse para destacar a parte do texto que você deseja comentar.
  3. Clique no pop-up **Add Feedback** que aparece abaixo do texto destacado.
  4. Siga as instruções apresentadas.
- Para enviar comentários mais complexos, crie um bilhete Bugzilla:
  1. Ir para o site da Bugzilla.
  2. Como Componente, use **Documentation**.
  3. Preencha o campo **Description** com sua sugestão de melhoria. Inclua um link para a(s) parte(s) relevante(s) da documentação.
  4. Clique em **Submit Bug**.

# CAPÍTULO 1. PLANEJANDO A IMPLANTAÇÃO DE UM SISTEMA DE ARQUIVOS GFS2

O sistema de arquivo Global File System 2 (GFS2) da Red Hat é um sistema de arquivo de cluster simétrico de 64 bits que fornece um espaço de nome compartilhado e gerencia a coerência entre múltiplos nós que compartilham um dispositivo de bloco comum. Um sistema de arquivo GFS2 destina-se a fornecer um conjunto de recursos que é o mais próximo possível de um sistema de arquivo local, ao mesmo tempo em que reforça a coerência total do cluster entre os nós. Para conseguir isso, os nós empregam um esquema de travamento em cluster para os recursos do sistema de arquivos. Este esquema de travamento utiliza protocolos de comunicação como o TCP/IP para trocar informações de travamento.

Em alguns casos, a API do sistema de arquivos Linux não permite que a natureza de cluster do GFS2 seja totalmente transparente; por exemplo, programas que utilizam bloqueios POSIX no GFS2 devem evitar o uso da função **GETLK** uma vez que, em um ambiente de cluster, a identificação do processo pode ser para um nó diferente no cluster. Na maioria dos casos, entretanto, a funcionalidade de um sistema de arquivos GFS2 é idêntica à de um sistema de arquivos local.

O Complemento de Armazenamento Resiliente do Red Hat Enterprise Linux (RHEL) fornece o GFS2, e depende do Complemento de Alta Disponibilidade RHEL para fornecer o gerenciamento de cluster exigido pelo GFS2.

O módulo do kernel **gfs2.ko** implementa o sistema de arquivos GFS2 e é carregado nos nós de cluster GFS2.

Para obter o melhor desempenho do GFS2, é importante levar em conta as considerações de desempenho que decorrem do projeto subjacente. Assim como um sistema de arquivo local, o GFS2 depende do cache de páginas para melhorar o desempenho através do cache local de dados freqüentemente utilizados. Para manter a coerência entre os nós no cluster, o controle do cache é fornecido pela máquina do estado *glock*.

**IMPORTANTE**

Certifique-se de que sua implementação do Red Hat High Availability Add-On atende às suas necessidades e pode ser apoiada. Consulte um representante autorizado da Red Hat para verificar sua configuração antes de sua implementação.

## 1.1. PARÂMETROS CHAVE GFS2 PARA DETERMINAR

Antes de instalar e configurar o GFS2, observe as seguintes características-chave de seus sistemas de arquivo GFS2:

**Nós GFS2**

Determinar quais nós do cluster irão montar os sistemas de arquivos GFS2.

**Número de sistemas de arquivo**

Determinar quantos sistemas de arquivo GFS2 devem ser criados inicialmente. Mais sistemas de arquivo podem ser adicionados posteriormente.

**Nome do sistema de arquivo**

Cada sistema de arquivo GFS2 deve ter um nome único. Este nome é normalmente o mesmo que o nome do volume lógico LVM e é usado como nome da tabela de bloqueio DLM quando um sistema de arquivo GFS2 é montado. Por exemplo, este guia usa nomes de sistemas de arquivos **mydata1** e **mydata2** em alguns exemplos de procedimentos.

**Periódicos**

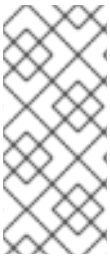
Determine o número de periódicos para seus sistemas de arquivos GFS2. O GFS2 requer um periódico para cada nó do cluster que precisa montar o sistema de arquivos. Por exemplo, se você tem um cluster de 16 nós mas precisa montar apenas o sistema de arquivos a partir de dois nós, você precisa apenas de dois periódicos. O GFS2 permite a adição dinâmica de periódicos em um ponto posterior com o utilitário **gfs2_jadd** como servidores adicionais montam um sistema de arquivos.

**Dispositivos de armazenamento e divisórias**

Determinar os dispositivos de armazenamento e partições a serem usados para criar volumes lógicos (usando **lvmlockd**) nos sistemas de arquivo.

**Protocolo de tempo**

Certifique-se de que os relógios nos nós GFS2 estejam sincronizados. É recomendado que você use o Protocolo de Tempo de Precisão (PTP) ou, se necessário para sua configuração, o software Network Time Protocol (NTP) fornecido com sua distribuição Red Hat Enterprise Linux.
Os relógios do sistema nos nós GFS2 devem estar a poucos minutos um do outro para evitar a atualização desnecessária do carimbo de tempo do inodo. A atualização desnecessária do carimbo de tempo inode afeta severamente o desempenho do cluster.



**NOTA**

Você pode ver problemas de desempenho com o GFS2 quando muitas operações de criação e exclusão são emitidas de mais de um nó no mesmo diretório ao mesmo tempo. Se isto causar problemas de desempenho em seu sistema, você deve localizar a criação e as exclusões de arquivos por um nó para diretórios específicos daquele nó, tanto quanto possível.

## 1.2. CONSIDERAÇÕES DE APOIO GFS2

Tabela 1.1, “GFS2 Limites de suporte” resume o tamanho máximo atual do sistema de arquivo e o número de nós que o GFS2 suporta.

**Tabela 1.1. GFS2 Limites de suporte**

| Parâmetro | Máximo |
|---|---|
| Número de nós | 16 (x86, Power8 on PowerVM)<br>4 (s390x sob z/VM) |
| Tamanho do sistema de arquivo | 100TB em todas as arquiteturas suportadas |

O GFS2 é baseado em uma arquitetura de 64 bits, que teoricamente pode acomodar um sistema de arquivos 8 EB. Se seu sistema requer sistemas de arquivo GFS2 maiores do que os atualmente suportados, entre em contato com seu representante de serviço da Red Hat.

NOTA

Embora um sistema de arquivo GFS2 possa ser implementado em um sistema autônomo ou como parte de uma configuração em cluster, a Red Hat não suporta o uso do GFS2 como um sistema de arquivo de nó único. A Red Hat suporta uma série de sistemas de arquivo de nó único de alto desempenho que são otimizados para nó único e, portanto, geralmente têm uma sobrecarga menor do que um sistema de arquivo de cluster. A Red Hat recomenda o uso destes sistemas de arquivo em preferência ao GFS2 nos casos em que apenas um único nó precisa montar o sistema de arquivo. Para informações sobre os sistemas de arquivo que o Red Hat Enterprise Linux 8 suporta, consulte Gerenciando sistemas de arquivo.

A Red Hat continuará a suportar sistemas de arquivo GFS2 de nó único para montagem de instantâneos de sistemas de arquivo de cluster, conforme seja necessário, por exemplo, para fins de backup.

Ao determinar o tamanho de seu sistema de arquivos, você deve considerar suas necessidades de recuperação. A execução do comando **fsck.gfs2** em um sistema de arquivo muito grande pode levar muito tempo e consumir uma grande quantidade de memória. Além disso, no caso de uma falha no disco ou subsistema de disco, o tempo de recuperação é limitado pela velocidade de sua mídia de backup. Para informações sobre a quantidade de memória que o comando **fsck.gfs2** requer, veja Determinar a memória necessária para executar o fsck.gfs2.

Enquanto um sistema de arquivo GFS2 pode ser usado fora do LVM, a Red Hat suporta apenas sistemas de arquivo GFS2 que são criados em um volume lógico LVM compartilhado.

NOTA

Quando você configura um sistema de arquivo GFS2 como um sistema de arquivo cluster, você deve garantir que todos os nós do cluster tenham acesso ao armazenamento compartilhado. Configurações assimétricas de cluster nas quais alguns nós têm acesso ao armazenamento compartilhado e outros não são suportados. Isto não requer que todos os nós realmente montem o próprio sistema de arquivo GFS2.

## 1.3. CONSIDERAÇÕES SOBRE A FORMATAÇÃO GFS2

Esta seção fornece recomendações sobre como formatar seu sistema de arquivos GFS2 para otimizar o desempenho.

IMPORTANTE

Certifique-se de que sua implementação do Red Hat High Availability Add-On atende às suas necessidades e pode ser apoiada. Consulte um representante autorizado da Red Hat para verificar sua configuração antes de sua implementação.

### Tamanho do sistema de arquivo: Mais pequeno é melhor

O GFS2 é baseado em uma arquitetura de 64 bits, que teoricamente pode acomodar um sistema de arquivos 8 EB. Entretanto, o tamanho máximo atualmente suportado de um sistema de arquivo GFS2 para hardware de 64 bits é de 100TB.

Observe que, embora os sistemas de arquivos grandes GFS2 sejam possíveis, isso não significa que eles sejam recomendados. A regra geral com GFS2 é que menor é melhor: é melhor ter sistemas de arquivo de 10 1TB do que um sistema de arquivo de 10TB.

Há várias razões pelas quais você deve manter seus sistemas de arquivos GFS2 pequenos:

- É necessário menos tempo para fazer o backup de cada sistema de arquivo.
- É necessário menos tempo se você precisar verificar o sistema de arquivo com o comando **fsck.gfs2**.
- Menos memória é necessária se você precisar verificar o sistema de arquivos com o comando **fsck.gfs2**.

Além disso, menos grupos de recursos para manter significam um melhor desempenho.

É claro, se você tornar seu sistema de arquivos GFS2 muito pequeno, você pode ficar sem espaço, e isso tem suas próprias conseqüências. Você deve considerar seus próprios casos de uso antes de decidir sobre um tamanho.

### Tamanho do bloco: Blocos padrão (4K) são preferidos

O comando **mkfs.gfs2** tenta estimar um tamanho de bloco ideal com base na topologia do dispositivo. Em geral, blocos 4K são o tamanho de bloco preferido porque 4K é o tamanho de página padrão (memória) para o Red Hat Enterprise Linux. Ao contrário de alguns outros sistemas de arquivo, o GFS2 faz a maioria de suas operações usando buffers de kernel 4K. Se seu tamanho de bloco for 4K, o kernel tem que fazer menos trabalho para manipular os buffers.

Recomenda-se usar o tamanho padrão do bloco, que deve render o mais alto desempenho. Você pode precisar usar um tamanho de bloco diferente somente se você precisar de um armazenamento eficiente de muitos arquivos muito pequenos.

### Tamanho da revista: Padrão (128MB) Normalmente é o ideal

Ao executar o comando **mkfs.gfs2** para criar um sistema de arquivos GFS2, você pode especificar o tamanho dos periódicos. Se você não especificar um tamanho, ele será padrão para 128MB, o que deve ser ideal para a maioria das aplicações.

Alguns administradores de sistema podem pensar que 128MB é excessivo e ser tentados a reduzir o tamanho do periódico para o mínimo de 8MB ou um 32MB mais conservador. Embora isso possa funcionar, pode ter um impacto severo no desempenho. Como muitos sistemas de arquivo de periódicos, toda vez que o GFS2 escreve metadados, os metadados são comprometidos com o periódico antes de ser colocado em funcionamento. Isto assegura que se o sistema falhar ou perder energia, você recuperará todos os metadados quando o periódico for reproduzido automaticamente no momento da montagem. Entretanto, não é necessária muita atividade do sistema de arquivos para preencher um diário de 8MB, e quando o diário está cheio, o desempenho diminui porque o GFS2 tem que esperar por escritos para o armazenamento.

Recomenda-se geralmente usar o tamanho padrão do diário de 128MB. Se seu sistema de arquivo for muito pequeno (por exemplo, 5GB), ter um diário de 128MB pode ser impraticável. Se você tiver um sistema de arquivo maior e puder arcar com o espaço, o uso de periódicos de 256MB pode melhorar o desempenho.

### Tamanho e número de grupos de recursos

Quando um sistema de arquivo GFS2 é criado com o comando **mkfs.gfs2**, ele divide o armazenamento em fatias uniformes conhecidas como grupos de recursos. Ele tenta estimar um tamanho ideal de grupo de recursos (variando de 32MB a 2GB). Você pode substituir o padrão com a opção **-r** do comando **mkfs.gfs2**.

O tamanho ideal de seu grupo de recursos depende de como você utilizará o sistema de arquivo. Considere quão cheio ele estará e se será ou não severamente fragmentado.

Você deve experimentar com diferentes tamanhos de grupos de recursos para ver qual resulta em um ótimo desempenho. É uma melhor prática experimentar com um grupo de teste antes de implantar o GFS2 na produção total.

Se seu sistema de arquivo tiver muitos grupos de recursos, cada um deles muito pequeno, a alocação de blocos pode perder muito tempo procurando dezenas de milhares de grupos de recursos por um bloco livre. Quanto mais cheio seu sistema de arquivo, mais grupos de recursos serão pesquisados, e cada um deles requer um bloqueio de cluster. Isto leva a um desempenho lento.

Se, no entanto, seu sistema de arquivo tiver muito poucos grupos de recursos, cada um dos quais é muito grande, as alocações de blocos podem ter mais freqüência para o mesmo bloqueio de grupo de recursos, o que também impacta o desempenho. Por exemplo, se você tem um sistema de arquivo de 10GB que é dividido em cinco grupos de recursos de 2GB, os nós em seu grupo lutarão mais frequentemente por esses cinco grupos de recursos do que se o mesmo sistema de arquivo fosse dividido em 320 grupos de recursos de 32MB. O problema é exacerbado se seu sistema de arquivos estiver quase cheio porque cada alocação de blocos pode ter que procurar em vários grupos de recursos antes de encontrar um com um bloco livre. O GFS2 tenta mitigar este problema de duas maneiras:

- Primeiro, quando um grupo de recursos está completamente cheio, ele se lembra disso e tenta evitar verificá-lo para futuras alocações até que um bloco seja liberado dele. Se você nunca apagar arquivos, a contenção será menos severa. Entretanto, se sua aplicação estiver constantemente apagando blocos e alocando novos blocos em um sistema de arquivos que esteja na maioria das vezes cheio, a contenção será muito alta e isto terá um impacto severo no desempenho.

- Segundo, quando novos blocos são adicionados a um arquivo existente (por exemplo, anexando) o GFS2 tentará agrupar os novos blocos no mesmo grupo de recursos que o arquivo. Isto é feito para aumentar o desempenho: em um disco giratório, as operações de busca demoram menos tempo quando estão fisicamente próximas entre si.

O pior cenário é quando existe um diretório central no qual todos os nós criam arquivos porque todos os nós lutarão constantemente para bloquear o mesmo grupo de recursos.

## 1.4. CONSIDERAÇÕES SOBRE O CLUSTER

Ao determinar o número de nós que seu sistema conterá, observe que existe um trade-off entre alta disponibilidade e desempenho. Com um número maior de nós, torna-se cada vez mais difícil fazer com que as cargas de trabalho aumentem. Por esse motivo, a Red Hat não suporta o uso do GFS2 para implantações de sistemas de arquivo de cluster maiores que 16 nós.

A implantação de um sistema de arquivo de cluster não é um substituto para a implantação de um único nó. A Red Hat recomenda que você permita um período de cerca de 8-12 semanas de testes em novas instalações a fim de testar o sistema e garantir que ele esteja trabalhando no nível de desempenho exigido. Durante este período, quaisquer questões de desempenho ou funcionais podem ser resolvidas e quaisquer dúvidas devem ser direcionadas à equipe de suporte da Red Hat.

A Red Hat recomenda que os clientes que considerem implantar clusters tenham suas configurações revisadas pelo suporte da Red Hat antes da implantação para evitar possíveis problemas de suporte posteriormente.

## 1.5. CONSIDERAÇÕES SOBRE FERRAGENS

Você deve levar em conta as seguintes considerações de hardware ao implantar um sistema de arquivos GFS2.

- Usar opções de armazenamento de maior qualidade
  O GFS2 pode operar com opções mais baratas de armazenamento compartilhado, como iSCSI ou Fibre Channel over Ethernet (FCoE), mas você obterá melhor desempenho se comprar um armazenamento de maior qualidade com maior capacidade de cache. A Red Hat realiza a maioria dos testes de qualidade, sanidade e desempenho em armazenamento SAN com interconexão Fibre Channel. Como regra geral, é sempre melhor implantar algo que tenha sido testado primeiro.

- Teste o equipamento da rede antes de implantar
  Equipamentos de rede de maior qualidade e mais rápidos fazem com que as comunicações em cluster e o GFS2 funcionem mais rapidamente com melhor confiabilidade. No entanto, não é necessário adquirir o hardware mais caro. Alguns dos switches de rede mais caros têm problemas para passar pacotes multicast, que são usados para passar os bloqueios (bandos) **fcntl**, enquanto os switches de rede mais baratos são às vezes mais rápidos e mais confiáveis. A Red Hat recomenda experimentar o equipamento antes de implementá-lo em plena produção.

# CAPÍTULO 2. RECOMENDAÇÕES PARA O USO DO GFS2

Esta seção fornece recomendações gerais sobre o uso do GFS2.

## 2.1. CONFIGURAÇÃO DE ATUALIZAÇÕES EM ATIME

Cada inode de arquivo e inode de diretório tem três carimbos de tempo associados a ele:

- **ctime**
- **mtime**
- **atime**

Se as atualizações do **atime** estiverem habilitadas como estão por padrão no GFS2 e em outros sistemas de arquivos Linux, então toda vez que um arquivo é lido, seu inode precisa ser atualizado.

Como poucas aplicações utilizam as informações fornecidas por **atime**, essas atualizações podem exigir uma quantidade significativa de tráfego de escrita desnecessária e tráfego de travamento de arquivos. Esse tráfego pode degradar o desempenho; portanto, pode ser preferível desligar ou reduzir a freqüência das atualizações de **atime**.

Os seguintes métodos para reduzir os efeitos da atualização do **atime** estão disponíveis:

- Monte com **relatime** (tempo relativo), que atualiza o **atime** se a atualização anterior **atime** for mais antiga que a atualização **mtime** ou **ctime**. Esta é a opção padrão de montagem para sistemas de arquivo GFS2.
- Monte com **noatime** ou **nodiratime**. A montagem com **noatime** desativa **atime** atualizações tanto para arquivos quanto para diretórios naquele sistema de arquivos, enquanto a montagem com **nodiratime** desativa **atime** atualizações somente para diretórios naquele sistema de arquivos, É geralmente recomendado montar os sistemas de arquivos GFS2 com a opção de montagem com **noatime** ou **nodiratime** sempre que possível, com a preferência para **noatime** onde a aplicação permite isso. Para maiores informações sobre o efeito destes argumentos no desempenho do sistema de arquivos GFS2, veja GFS2 Node Locking.

Use o seguinte comando para montar um sistema de arquivo GFS2 com a opção de montagem em **noatime** Linux.

```
monte BlockDevice MountPoint -o noatime
```

**BlockDevice**

Especifica o dispositivo de bloco onde reside o sistema de arquivos GFS2.

**MountPoint**

Especifica o diretório onde o sistema de arquivos GFS2 deve ser montado.

Neste exemplo, o sistema de arquivos GFS2 reside em **/dev/vg01/lvol0** e é montado no diretório **/mygfs2** com **atime** atualizações desativadas.

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

## 2.2. OPÇÕES DE SINTONIA VFS: PESQUISA E EXPERIÊNCIA

Como todos os sistemas de arquivo Linux, o GFS2 está sobre uma camada chamada sistema de arquivo virtual (VFS). O VFS fornece bons padrões para as configurações de cache para a maioria das cargas de trabalho e não deve precisar ser alterado na maioria dos casos. Se, no entanto, você tiver uma carga de trabalho que não esteja rodando eficientemente (por exemplo, o cache é muito grande ou muito pequeno), então você poderá melhorar o desempenho usando o comando **sysctl**(8) para ajustar os valores dos arquivos **sysctl** no diretório /**proc**/**sys**/**vm**. A documentação para estes arquivos pode ser encontrada na árvore de fontes do kernel **Documentation**/**sysctl**/**vm.txt**.

Por exemplo, os valores para **dirty_background_ratio** e **vfs_cache_pressure** podem ser ajustados, dependendo de sua situação. Para obter os valores atuais, use os seguintes comandos:

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

Os seguintes comandos ajustam os valores:

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

Você pode alterar permanentemente os valores destes parâmetros editando o arquivo /**etc**/**sysctl.conf**.

Para encontrar os valores ideais para seus casos de uso, pesquise as várias opções de VFS e experimente em um grupo de teste antes de implantar em produção plena.

## 2.3. SELINUX NO GFS2

O uso do Security Enhanced Linux (SELinux) com GFS2 incorre em uma pequena penalidade de desempenho. Para evitar esta sobrecarga, você pode optar por não utilizar o SELinux com GFS2 mesmo em um sistema com SELinux em modo de aplicação. Ao montar um sistema de arquivo GFS2, você pode garantir que o SELinux não tentará ler o elemento **seclabel** em cada objeto do sistema de arquivo usando uma das opções **context**, conforme descrito na página de manual **mount**(8); o SELinux assumirá que todo o conteúdo do sistema de arquivo está etiquetado com o elemento **seclabel** fornecido nas opções de montagem **context**. Isto também agilizará o processamento, pois evita outra leitura em disco do bloco de atributos estendido que poderia conter elementos **seclabel**.

Por exemplo, em um sistema com SELinux em modo de aplicação, você pode usar o seguinte comando **mount** para montar o sistema de arquivos GFS2 se o sistema de arquivos for conter conteúdo Apache. Esta etiqueta será aplicada a todo o sistema de arquivo; ela permanece na memória e não é gravada em disco.

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

Se você não tiver certeza se o sistema de arquivo conterá conteúdo Apache, você pode usar os rótulos **public_content_rw_t** ou **public_content_t**, ou você pode definir um novo rótulo e definir uma política em torno dele.

Observe que em um cluster Pacemaker você deve sempre usar Pacemaker para gerenciar um sistema de arquivos GFS2. Você pode especificar as opções de montagem ao criar um recurso de sistema de arquivo GFS2.

## 2.4. CONFIGURANDO O NFS SOBRE O GFS2

Devido à complexidade adicional do subsistema de travamento GFS2 e sua natureza agrupada, a

configuração do NFS sobre o GFS2 requer tomar muitas precauções e uma configuração cuidadosa. Esta seção descreve as advertências que você deve levar em conta ao configurar um serviço NFS sobre um sistema de arquivos GFS2.

**ATENÇÃO**

Se o sistema de arquivo GFS2 é NFS exportado, então você deve montar o sistema de arquivo com a opção **localflocks**. Como a utilização da opção **localflocks** impede o acesso seguro ao sistema de arquivos GFS2 de múltiplos locais, e não é viável exportar o GFS2 de múltiplos nós simultaneamente, é um requisito de suporte que o sistema de arquivos GFS2 seja montado em apenas um nó de cada vez ao utilizar esta configuração. O efeito pretendido disto é forçar os bloqueios POSIX de cada servidor a serem locais: não exclusivos, independentes uns dos outros. Isto ocorre porque existe uma série de problemas se o GFS2 tentar implementar bloqueios POSIX a partir do NFS através dos nós de um cluster. Para aplicações rodando em clientes NFS, fechaduras POSIX localizadas significam que dois clientes podem segurar a mesma fechadura simultaneamente se os dois clientes forem montados a partir de servidores diferentes, o que poderia causar corrupção de dados. Se todos os clientes montam NFS a partir de um servidor, então o problema de servidores separados concedendo os mesmos bloqueios independentemente vai embora. Se você não tiver certeza se deve montar seu sistema de arquivos com a opção **localflocks**, você não deve usar a opção. Contate imediatamente o suporte da Red Hat para discutir a configuração apropriada a fim de evitar a perda de dados. A exportação do GFS2 via NFS, embora tecnicamente suportada em algumas circunstâncias, não é recomendada.

Para todas as outras aplicações (não-NFS) GFS2, não monte seu sistema de arquivos usando **localflocks**, para que o GFS2 gerencie os bloqueios POSIX e os bandos entre todos os nós do cluster (em uma base de cluster). Se você especificar **localflocks** e não usar NFS, os outros nós do aglomerado não terão conhecimento dos bloqueios e bandos POSIX uns dos outros, tornando-os assim inseguros em um ambiente agrupado

Além das considerações de bloqueio, você deve levar em conta o seguinte ao configurar um serviço NFS sobre um sistema de arquivos GFS2.

- A Red Hat suporta apenas as configurações do Red Hat High Availability Add-On usando NFSv3 com travamento em uma configuração ativa/passiva com as seguintes características. Esta configuração fornece Alta Disponibilidade (HA) para o sistema de arquivo e reduz o tempo de inatividade do sistema, já que um nó com falha não resulta na necessidade de executar o comando **fsck** quando falha o servidor NFS de um nó para outro.
    - O sistema de arquivo back-end é um sistema de arquivo GFS2 que roda em um cluster de 2 a 16 nós.
    - Um servidor NFSv3 é definido como um serviço que exporta todo o sistema de arquivos GFS2 a partir de um único nó de cluster de cada vez.
    - O servidor NFS pode falhar de um nó de cluster para outro (configuração ativa/passiva).
    - Nenhum acesso ao sistema de arquivos GFS2 é permitido *except* através do servidor NFS.

Isto inclui tanto o acesso ao sistema de arquivos local GFS2 quanto o acesso através do Samba ou Clustered Samba. O acesso ao sistema de arquivo localmente através do nó de cluster a partir do qual ele é montado pode resultar em corrupção de dados.

- Não há suporte de cota NFS no sistema.

- A opção **fsid=** NFS é obrigatória para as exportações NFS de GFS2.

- Se surgirem problemas com seu agrupamento (por exemplo, o agrupamento se torna inquieto e a vedação não é bem sucedida), os volumes lógicos do agrupamento e o sistema de arquivos GFS2 serão congelados e nenhum acesso será possível até que o agrupamento seja quorato. Você deve considerar esta possibilidade ao determinar se uma simples solução de failover, como a definida neste procedimento, é a mais apropriada para seu sistema.

## 2.5. ARQUIVO SAMBA (SMB OU WINDOWS) SERVINDO SOBRE O GFS2

Você pode usar arquivos Samba (SMB ou Windows) servindo a partir de um sistema de arquivos GFS2 com CTDB, o que permite configurações ativas/atuantes.

O acesso simultâneo aos dados no Samba share a partir de fora do Samba não é suportado. Atualmente não há suporte para os aluguéis do cluster GFS2, o que retarda o serviço de arquivos do Samba. Para maiores informações sobre políticas de suporte para Samba, veja Políticas de Suporte para Armazenamento Resiliente RHEL - ctdb Políticas Gerais e Políticas de Suporte para Armazenamento Resiliente RHEL - Exportação de conteúdo gfs2 através de outros protocolos no Portal do Cliente da Red Hat.

## 2.6. CONFIGURAÇÃO DE MÁQUINAS VIRTUAIS PARA GFS2

Ao utilizar um sistema de arquivo GFS2 com uma máquina virtual, é importante que suas configurações de armazenamento de VM em cada nó sejam configuradas corretamente para forçar o cache a ser desligado. Por exemplo, incluir estas configurações para **cache** e **io** no domínio **libvirt** deve permitir que o GFS2 se comporte como esperado.

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

Alternativamente, você pode configurar o atributo **shareable** dentro do elemento do dispositivo. Isto indica que o dispositivo deve ser compartilhado entre domínios (desde que o hypervisor e o SO suportem isto). Se **shareable** for usado, **cache='no'** deve ser usado para esse dispositivo.

## 2.7. ALOCAÇÃO EM BLOCO

Esta seção fornece um resumo das questões relacionadas à alocação de blocos nos sistemas de arquivos GFS2. Mesmo que aplicações que só escrevem dados normalmente não se importam como ou onde um bloco é alocado, algum conhecimento de como funciona a alocação de blocos pode ajudá-lo a otimizar o desempenho.

### 2.7.1. Deixar espaço livre no sistema de arquivo

Quando um sistema de arquivo GFS2 está quase cheio, o alocador de blocos começa a ter dificuldades para encontrar espaço para novos blocos a serem alocados. Como resultado, os blocos distribuídos pelo alocador tendem a ser espremidos no final de um grupo de recursos ou em fatias minúsculas onde a fragmentação do arquivo é muito mais provável. Esta fragmentação de arquivo pode causar problemas de desempenho. Além disso, quando um sistema de arquivo GFS2 está quase cheio, o alocador de

blocos GFS2 gasta mais tempo pesquisando em vários grupos de recursos, e isso acrescenta a contenção de bloqueios que não necessariamente existiriam em um sistema de arquivo que tem amplo espaço livre. Isto também pode causar problemas de desempenho.

Por estas razões, é recomendável que você não execute um sistema de arquivo que esteja mais de 85% cheio, embora este número possa variar dependendo da carga de trabalho.

### 2.7.2. Fazer com que cada nó aloque seus próprios arquivos, se possível

Ao desenvolver aplicações para uso com sistemas de arquivos GFS2, é recomendável que cada nó atribua seus próprios arquivos, se possível. Devido à forma como o gerenciador de fechaduras distribuídas (DLM) funciona, haverá mais contenção de fechaduras se todos os arquivos forem alocados por um nó e outros nós precisarem adicionar blocos a esses arquivos.

Com o DLM, o primeiro nó a bloquear um recurso (como um arquivo) torna-se o "mestre da fechadura" para aquela fechadura. Outros nós podem bloquear esse recurso, mas eles têm que pedir permissão ao mestre de fechaduras primeiro. Cada nó sabe que fechaduras para as quais é o mestre de fechaduras, e cada nó sabe a que nó emprestou uma fechadura. Bloquear uma fechadura no nó mestre é muito mais rápido que bloquear uma fechadura em outro nó que tem que parar e pedir permissão ao mestre da fechadura.

Como em muitos sistemas de arquivo, o alocador GFS2 tenta manter os blocos no mesmo arquivo próximos uns dos outros para reduzir o movimento das cabeças de disco e aumentar o desempenho. Um nó que aloca blocos a um arquivo provavelmente precisará usar e bloquear os mesmos grupos de recursos para os novos blocos (a menos que todos os blocos desse grupo de recursos estejam em uso). O sistema de arquivo será executado mais rapidamente se o mestre de bloqueio para o grupo de recursos que contém o arquivo alocar seus blocos de dados (é mais rápido ter o nó que primeiro abriu o arquivo fazendo toda a escrita dos novos blocos).

### 2.7.3. Pré-atribuir, se possível

Se os arquivos forem pré-alocados, as alocações em bloco podem ser evitadas completamente e o sistema de arquivos pode funcionar de forma mais eficiente. O GFS2 inclui a chamada ao sistema **fallocate(1)**, que pode ser usada para pré-alocação de blocos de dados.

# CAPÍTULO 3. SISTEMAS DE ARQUIVO GFS2

Esta seção fornece informações sobre os comandos e opções que você usa para criar, montar e expandir os sistemas de arquivos GFS2.

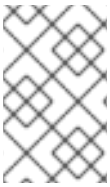
## 3.1. CRIAÇÃO DO SISTEMA DE ARQUIVOS GFS2

Você cria um sistema de arquivos GFS2 com o comando **mkfs.gfs2**. Um sistema de arquivo é criado em um volume LVM ativado.

### 3.1.1. O comando GFS2 mkfs

As seguintes informações são necessárias para executar o comando **mkfs.gfs2** para criar um sistema de arquivos GFS2 em cluster:

- Nome do protocolo/módulo de bloqueio, que é **lock_dlm** para um cluster
- Nome do grupo
- Número de periódicos (um diário necessário para cada nó que pode estar montando o sistema de arquivo)



**NOTA**

Uma vez criado um sistema de arquivo GFS2 com o comando **mkfs.gfs2**, não se pode diminuir o tamanho do sistema de arquivo. Você pode, entretanto, aumentar o tamanho de um sistema de arquivo existente com o comando **gfs2_grow**.

O formato para criar um sistema de arquivo GFS2 agrupado é o seguinte. Note que a Red Hat não suporta o uso do GFS2 como um sistema de arquivo de nó único.

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

Se preferir, você pode criar um sistema de arquivo GFS2 usando o comando **mkfs** com o parâmetro **-t** especificando um sistema de arquivo do tipo **gfs2**, seguido das opções do sistema de arquivo GFS2.

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

**ATENÇÃO**

Especificar inadequadamente o parâmetro *ClusterName:FSName* pode causar corrupção do sistema de arquivos ou do espaço de bloqueio.

**ClusterName**

O nome do cluster para o qual o sistema de arquivos GFS2 está sendo criado.

**FSName**

O nome do sistema de arquivo, que pode ter de 1 a 16 caracteres. O nome deve ser único para todos os sistemas de arquivo **lock_dlm** sobre o cluster.

**NumberJournals**

Especifica o número de periódicos a serem criados pelo comando **mkfs.gfs2**. É necessário um periódico para cada nó que monta o sistema de arquivos. Para sistemas de arquivo GFS2, mais periódicos podem ser adicionados posteriormente sem aumentar o sistema de arquivo.

**BlockDevice**

Especifica um dispositivo lógico ou outro dispositivo de bloco

Tabela 3.1, "Opções de comando **mkfs.gfs2**" descreve as opções de comando **mkfs.gfs2** (bandeiras e parâmetros).

**Tabela 3.1. Opções de comando mkfs.gfs2**

| Bandeira | Parâmetro | Descrição |
|---|---|---|
| **-c** | **Megabytes** | Define o tamanho inicial do arquivo de alteração de cota de cada revista para **Megabytes**. |
| **-D** | | Permite a depuração da saída. |
| **-h** | | Ajuda. Exibe as opções disponíveis. |
| **-J** | **Megabytes** | Especifica o tamanho do periódico em megabytes. O tamanho padrão do periódico é de 128 megabytes. O tamanho mínimo é de 8 megabytes. Periódicos maiores melhoram o desempenho, embora usem mais memória do que os periódicos menores. |
| **-j** | **Number** | Especifica o número de periódicos a serem criados pelo comando **mkfs.gfs2**. É necessário um periódico para cada nó que monta o sistema de arquivos. Se esta opção não for especificada, será criado um periódico. Para os sistemas de arquivo GFS2, é possível adicionar mais periódicos em um momento posterior sem aumentar o sistema de arquivo. |
| **-O** | | Impede que o comando **mkfs.gfs2** peça confirmação antes de escrever o sistema de arquivo. |

| Bandeira | Parâmetro | Descrição |
|---|---|---|
| **-p** | **LockProtoName** | * Especifica o nome do protocolo de travamento a ser utilizado. Os protocolos de travamento reconhecidos incluem:<br>* **lock_dlm**<br>* **lock_nolock** |
| **-q** | | Silencioso. Não exiba nada. |
| **-r** | **Megabytes** | Especifica o tamanho dos grupos de recursos em megabytes. O tamanho mínimo do grupo de recursos é de 32 megabytes. O tamanho máximo do grupo de recursos é de 2048 megabytes. Um tamanho grande de grupo de recursos pode aumentar o desempenho em sistemas de arquivo muito grandes. Se isto não for especificado, **mkfs.gfs2** escolhe o tamanho do grupo de recursos com base no tamanho do sistema de arquivos: sistemas de arquivos de tamanho médio terão 256 megabytes de grupos de recursos, e sistemas de arquivos maiores terão RGs maiores para melhor desempenho. |

| Bandeira | Parâmetro | Descrição |
|---|---|---|
| **-t** | **LockTableName** | * Um identificador único que especifica o campo da tabela de bloqueio quando você usa o protocolo **lock_dlm**; o protocolo **lock_nolock** não usa este parâmetro.<br><br>* Este parâmetro tem duas partes separadas por dois pontos (sem espaços) como se segue: **ClusterName:FSName** .<br><br>* **ClusterName** é o nome do cluster para o qual o sistema de arquivos GFS2 está sendo criado; somente os membros deste cluster estão autorizados a utilizar este sistema de arquivos.<br><br>* **FSName** , o nome do sistema de arquivo, pode ter de 1 a 16 caracteres de comprimento, e o nome deve ser único entre todos os sistemas de arquivo do cluster. |
| **-V** | | Exibe as informações da versão de comando. |

### 3.1.2. Criação de um sistema de arquivos GFS2

O exemplo a seguir cria dois sistemas de arquivo GFS2. Para estes dois sistemas de arquivo, lock_dlm` é o protocolo de travamento que o sistema de arquivo utiliza, uma vez que este é um sistema de arquivo agrupado. Ambos os sistemas de arquivo podem ser usados no cluster chamado **alpha**.

Para o primeiro sistema de arquivo, o nome do sistema de arquivo é **mydata1**. ele contém oito periódicos e é criado em **/dev/vg01/lvol0**. Para o segundo sistema de arquivo, o nome do sistema de arquivo é **mydata2**. Ele contém oito periódicos e é criado em **/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. MONTAGEM DE UM SISTEMA DE ARQUIVO GFS2

NOTA

Você deve sempre usar o Pacemaker para gerenciar o sistema de arquivos GFS2 em um ambiente de produção em vez de montar manualmente o sistema de arquivos com um comando **mount**, pois isso pode causar problemas no desligamento do sistema, conforme descrito em Desmontagem de um sistema de arquivos GFS2 .

Antes que você possa montar um sistema de arquivo GFS2, o sistema de arquivo deve existir, o volume onde o sistema de arquivo existe deve ser ativado, e os sistemas de agrupamento e travamento de suporte devem ser iniciados. Após atender a esses requisitos, você pode montar o sistema de arquivo GFS2 como faria com qualquer sistema de arquivo Linux.

Para o bom funcionamento do sistema de arquivos GFS2, o pacote **gfs2-utils** deve ser instalado em todos os nós que montam um sistema de arquivos GFS2. O pacote **gfs2-utils** faz parte do canal de armazenamento resiliente.

Para manipular ACLs de arquivo, você deve montar o sistema de arquivo com a opção de montagem **-o acl**. Se um sistema de arquivo for montado sem a opção de montagem **-o acl**, os usuários podem visualizar as ACLs (com **getfacl**), mas não é permitido defini-las (com **setfacl**).

### 3.2.1. Montagem de um sistema de arquivo GFS2 sem opções especificadas

Neste exemplo, o sistema de arquivos GFS2 em **/dev/vg01/lvol0** está montado no diretório **/mygfs2**.

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

### 3.2.2. Montagem de um sistema de arquivo GFS2 que especifica as opções de montagem

A seguir está o formato do comando para montar um sistema de arquivo GFS2 que especifica as opções de montagem.

```
montar BlockDevice MountPoint -o option
```

**BlockDevice**

Especifica o dispositivo de bloco onde reside o sistema de arquivos GFS2.

**MountPoint**

Especifica o diretório onde o sistema de arquivos GFS2 deve ser montado.

O argumento **-o option** consiste em opções específicas GFS2 (ver Tabela 3.2, "GFS2 - Opções de Montagem específicas") ou opções padrão aceitáveis do Linux **mount -o**, ou uma combinação de ambas. Vários parâmetros **option** são separados por uma vírgula e sem espaços.

NOTA

O comando **mount** é um comando de sistema Linux. Além de usar as opções específicas do GFS2 descritas nesta seção, você pode usar outras opções de comando, padrão **mount** (por exemplo, **-r**). Para informações sobre outras opções de comando do Linux **mount**, consulte a página de manual do Linux **mount**.

Tabela 3.2, "GFS2 - Opções de Montagem específicas" descreve os valores disponíveis do GFS2 específico **-o option** que podem ser passados ao GFS2 no momento da montagem.

NOTA

Esta tabela inclui descrições de opções que são usadas somente com sistemas de arquivo locais. Note, entretanto, que a Red Hat não suporta o uso do GFS2 como um sistema de arquivo de nó único. A Red Hat continuará a suportar sistemas de arquivo GFS2 de nó único para montagem de instantâneos de sistemas de arquivo cluster (por exemplo, para fins de backup).

Tabela 3.2. GFS2 - Opções de Montagem específicas

| Opção | Descrição |
|---|---|
| **acl** | Permite manipular ACLs de arquivos. Se um sistema de arquivo é montado sem a opção de montagem **acl**, os usuários podem visualizar as ACLs (com **getfacl**), mas não têm permissão para configurá-las (com **setfacl**). |
| **data=[ordered\|writeback]** | Quando **data=ordered** é definido, os dados do usuário modificados por uma transação são enxaguados para o disco antes que a transação seja comprometida com o disco. Isto deve evitar que o usuário veja blocos não inicializados em um arquivo após uma falha. Quando o modo **data=writeback** é configurado, os dados do usuário são gravados no disco a qualquer momento depois de serem sujos; isto não oferece a mesma garantia de consistência do modo **ordered**, mas deve ser um pouco mais rápido para algumas cargas de trabalho. O valor padrão é o modo **ordered**. |
| * **ignore_local_fs**<br><br>* **Caution:** Esta opção deve ser usada*not* quando os sistemas de arquivo GFS2 são compartilhados. | Força o GFS2 a tratar o sistema de arquivos como um sistema de arquivos multi-hospedeiro. Por padrão, usando **lock_nolock** acende automaticamente a bandeira **localflocks**. |
| * **localflocks**<br><br>* **Caution:** Esta opção não deve ser utilizada quando os sistemas de arquivo GFS2 são compartilhados. | Diz ao GFS2 para deixar a camada VFS (sistema de arquivo virtual) fazer todo o rebanho e fcntl. A bandeira **localflocks** é automaticamente ativada por **lock_nolock**. |
| **lockproto=LockModuleName** | Permite ao usuário especificar qual protocolo de travamento deve ser usado com o sistema de arquivo. Se **LockModuleName** não for especificado, o nome do protocolo de travamento é lido a partir do superbloco do sistema de arquivo. |
| **locktable=LockTableName** | Permite que o usuário especifique qual tabela de bloqueio deve ser usada com o sistema de arquivo. |

| Opção | Descrição |
| --- | --- |
| **quota=[off/account/on]** | Liga ou desliga as cotas para um sistema de arquivo. A definição das cotas no estado **account** faz com que as estatísticas de uso por UID/GID sejam corretamente mantidas pelo sistema de arquivo; os valores limite e de advertência são ignorados. O valor padrão é **off**. |
| **errors=panic\|withdraw** | Quando **errors=panic** é especificado, os erros do sistema de arquivo causarão pânico no núcleo. Quando for especificado **errors=withdraw**, que é o comportamento padrão, os erros do sistema de arquivo causarão a retirada do sistema de arquivo e o tornarão inacessível até a próxima reinicialização; em alguns casos, o sistema pode permanecer em funcionamento. |
| **discard/nodiscard** | Faz com que o GFS2 gere pedidos de "descarte" de E/S para blocos que foram liberados. Estes podem ser usados por hardware adequado para implementar esquemas de provisionamento fino e similares. |
| **barrier/nobarrier** | Faz com que o GFS2 envie barreiras de E/S ao descarregar a revista. O valor padrão é **on**. Esta opção é automaticamente girada **off** se o dispositivo subjacente não suportar as barreiras de E/S. O uso de barreiras de E/S com GFS2 é altamente recomendado em todos os momentos, a menos que o dispositivo de bloqueio seja projetado de forma que não possa perder seu conteúdo de cache de escrita (por exemplo, se estiver em um UPS ou se não tiver um cache de escrita). |
| **quota_quantum=*secs*** | Define o número de segundos durante os quais uma alteração nas informações de cota pode ser feita em um nó antes de ser escrita no arquivo de cota. Esta é a forma preferida para definir este parâmetro. O valor é um número inteiro de segundos maior que zero. O valor padrão é 60 segundos. Ajustes mais curtos resultam em atualizações mais rápidas das informações de cota preguiçosas e menor probabilidade de alguém exceder sua cota. Configurações mais longas tornam as operações do sistema de arquivos envolvendo cotas mais rápidas e mais eficientes. |

| Opção | Descrição |
| --- | --- |
| **statfs_quantum=*secs*** | Definir **statfs_quantum** para 0 é a forma preferida de definir a versão lenta de **statfs**. O valor padrão é 30 segundos, o que define o período máximo de tempo antes que **statfs** as mudanças sejam sincronizadas com o arquivo principal **statfs**. Isto pode ser ajustado para permitir valores **statfs** mais rápidos e menos precisos ou valores mais lentos e precisos. Quando esta opção for definida como 0, **statfs** sempre relatará os valores verdadeiros. |
| **statfs_percent=*value*** | Fornece um limite sobre a porcentagem máxima de mudança nas informações do **statfs** em uma base local antes de ser sincronizado de volta ao arquivo principal **statfs**, mesmo que o período de tempo não tenha expirado. Se a configuração de **statfs_quantum** for 0, então esta configuração é ignorada. |

### 3.2.3. Desmontando um sistema de arquivo GFS2

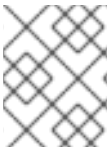
Os sistemas de arquivo GFS2 que foram montados manualmente em vez de automaticamente através do Pacemaker não serão conhecidos pelo sistema quando os sistemas de arquivo forem desmontados no desligamento do sistema. Como resultado, o agente de recursos GFS2 não desmontará o sistema de arquivos GFS2. Após o desligamento do agente de recurso GFS2, o processo padrão de desligamento mata todos os processos restantes do usuário, incluindo a infra-estrutura de cluster, e tenta desmontar o sistema de arquivo. Esta desmontagem falhará sem a infra-estrutura do cluster e o sistema ficará pendurado.

Para evitar que o sistema fique pendurado quando os sistemas de arquivo GFS2 não estiverem montados, você deve fazer uma das seguintes ações:

- Sempre use Pacemaker para gerenciar o sistema de arquivos GFS2.
- Se um sistema de arquivo GFS2 tiver sido montado manualmente com o comando **mount**, certifique-se de desmontar o sistema de arquivo manualmente com o comando **umount** antes de reiniciar ou desligar o sistema.

Se seu sistema de arquivo estiver pendurado enquanto estiver sendo desmontado durante o desligamento do sistema sob estas circunstâncias, faça uma reinicialização do hardware. É improvável que qualquer dado seja perdido, já que o sistema de arquivos é sincronizado mais cedo no processo de desligamento.

O sistema de arquivos GFS2 pode ser desmontado da mesma forma que qualquer sistema de arquivos Linux, usando o comando **umount**.



NOTA

O comando **umount** é um comando de sistema Linux. Informações sobre este comando podem ser encontradas nas páginas de manual de comando do Linux **umount**.

Utilização

```
umount MountPoint
```

**MountPoint**

Especifica o diretório onde o sistema de arquivos GFS2 está atualmente montado.

## 3.3. CÓPIA DE SEGURANÇA DE UM SISTEMA DE ARQUIVO GFS2

É importante fazer backups regulares de seu sistema de arquivos GFS2 em caso de emergência, independentemente do tamanho de seu sistema de arquivos. Muitos administradores de sistema se sentem seguros porque são protegidos por RAID, multicaminhos, espelhamento, instantâneos e outras formas de redundância, mas não existe tal coisa como seguro o suficiente.

Pode ser um problema criar um backup, já que o processo de backup de um nó ou conjunto de nós geralmente envolve a leitura de todo o sistema de arquivos em seqüência. Se isso for feito a partir de um único nó, esse nó reterá todas as informações no cache até que outros nós do cluster comecem a solicitar bloqueios. A execução deste tipo de programa de backup enquanto o cluster estiver em operação terá um impacto negativo no desempenho.

Soltar as caches uma vez que o backup esteja completo reduz o tempo exigido pelos outros nós para recuperar a propriedade de suas fechaduras/caches de cluster. Isto ainda não é ideal, no entanto, porque os outros nós terão parado de armazenar em cache os dados que estavam armazenando antes do início do processo de backup. Você pode soltar os caches usando o seguinte comando depois que o backup estiver completo:

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

É mais rápido se cada nó do cluster fizer backup de seus próprios arquivos para que a tarefa seja dividida entre os nós. Você pode conseguir isso com um script que usa o comando **rsync** em diretórios específicos de node-specific directories.

A Red Hat recomenda fazer um backup GFS2 criando um instantâneo de hardware na SAN, apresentando o instantâneo para outro sistema, e fazendo o backup lá em cima. O sistema de backup deve montar o snapshot com **-o lockproto=lock_nolock**, já que ele não estará em um cluster.

## 3.4. SUSPENDER A ATIVIDADE EM UM SISTEMA DE ARQUIVOS GFS2

Você pode suspender a atividade de gravação em um sistema de arquivo usando o comando **dmsetup suspend**. Suspender a atividade de gravação permite que instantâneos de dispositivos baseados em hardware sejam usados para capturar o sistema de arquivo em um estado consistente. O comando **dmsetup resume** termina a suspensão.

O formato para o comando de suspensão de atividade em um sistema de arquivos GFS2 é o seguinte.

```
dmsetup suspender MountPoint
```

Este exemplo suspende a escrita no sistema de arquivo /**mygfs2**.

```
# dmsetup suspend /mygfs2
```

O formato para o comando para terminar a suspensão de atividade em um sistema de arquivo GFS2 é o seguinte.

```
currículo dmsetup MountPoint
```

Este exemplo acaba com a suspensão dos escritos ao sistema de arquivo /**mygfs2**.

```
# dmsetup resume /mygfs2
```

## 3.5. CULTIVO DE UM SISTEMA DE ARQUIVO GFS2

O comando **gfs2_grow** é usado para expandir um sistema de arquivo GFS2 após o dispositivo onde o sistema de arquivo reside ter sido expandido. A execução do comando **gfs2_grow** em um sistema de arquivo GFS2 existente preenche todo o espaço livre entre a extremidade atual do sistema de arquivo e o final do dispositivo com uma extensão do sistema de arquivo GFS2 recém-inicializada. Todos os nós do cluster podem então usar o espaço de armazenamento extra que foi adicionado.

NOTA

Você não pode diminuir o tamanho de um sistema de arquivo GFS2.

O comando **gfs2_grow** deve ser executado em um sistema de arquivo montado. O seguinte procedimento aumenta o tamanho do sistema de arquivo GFS2 em um cluster que é montado no volume lógico **shared_vg**/**shared_lv1** com um ponto de montagem de /**mnt**/**gfs2**.

1. Realizar um backup dos dados no sistema de arquivos.
2. Se você não sabe o volume lógico que é usado pelo sistema de arquivo a ser expandido, você pode determinar isso executando o **df** ***mountpoint*** comando. Isto exibirá o nome do dispositivo no seguinte formato:
   /**dev**/**mapper**/***vg-lv***

   Por exemplo, o nome do dispositivo /**dev**/**mapper**/**shared_vg-shared_lv1** indica que o volume lógico é **shared_vg**/**shared_lv1**.
3. Em um nó do cluster, expandir o volume subjacente do cluster com o comando **lvextend**, usando a opção **--lockopt skiplv** para anular o bloqueio lógico normal do volume.

   ```
   # lvextend --lockopt skiplv -L+1G shared_vg/shared_lv1
   WARNING: skipping LV lock in lvmlockd.
   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. Se você estiver executando o RHEL 8.0, em cada nó adicional do cluster atualize o volume lógico para atualizar o volume lógico ativo naquele nó. Esta etapa não é necessária em sistemas rodando RHEL 8.1 e mais tarde, pois a etapa é automatizada quando o volume lógico é estendido.

   ```
   # lvchange --refresh shared_vg/shared_lv1
   ```
5. Um nó do cluster, aumenta o tamanho do sistema de arquivos GFS2. Não estender o sistema de arquivo se o volume lógico não foi atualizado em todos os nós, caso contrário, os dados do sistema de arquivo podem ficar indisponíveis em todo o cluster.

   ```
   # gfs2_grow /mnt/gfs2
   FS: Mount point:            /mnt/gfs2
   ```

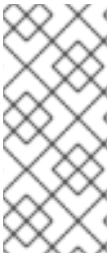
```
FS: Device:                /dev/mapper/shared_vg-shared_lv1
FS: Size:                  1310719 (0x13ffff)
DEV: Length:               1572864 (0x180000)
The file system will grow by 1024MB.
gfs2_grow complete.
```

6. Execute o comando **df** em todos os nós para verificar se o novo espaço está agora disponível no sistema de arquivos. Note que pode levar até 30 segundos para o comando **df** em todos os nós mostrar o mesmo tamanho de sistema de arquivo

```
# 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. ADICIONANDO PERIÓDICOS A UM SISTEMA DE ARQUIVOS GFS2

O GFS2 requer um diário para cada nó em um cluster que precisa montar o sistema de arquivo. Se você adicionar nós adicionais ao cluster, você pode adicionar periódicos a um sistema de arquivos GFS2 com o comando **gfs2_jadd**. Você pode adicionar periódicos a um sistema de arquivos GFS2 dinamicamente em qualquer ponto sem expandir o volume lógico subjacente. O comando **gfs2_jadd** deve ser executado em um sistema de arquivo montado, mas ele precisa ser executado em apenas um nó no cluster. Todos os outros nós sentem que a expansão ocorreu.



NOTA

Se um sistema de arquivo GFS2 estiver cheio, o comando **gfs2_jadd** falhará, mesmo que o volume lógico contendo o sistema de arquivo tenha sido estendido e seja maior do que o sistema de arquivo. Isto porque em um sistema de arquivos GFS2, os periódicos são arquivos simples ao invés de metadados embutidos, portanto a simples extensão do volume lógico subjacente não fornecerá espaço para os periódicos.

Antes de adicionar periódicos a um sistema de arquivos GFS2, você pode descobrir quantos periódicos o sistema de arquivos GFS2 contém atualmente com o comando **gfs2_edit -p jindex**, como no exemplo a seguir:

```
# 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
```

O formato para o comando básico para adicionar periódicos a um sistema de arquivos GFS2 é o seguinte.

```
gfs2_jadd -j Number MountPoint
```

**Number**

Especifica o número de novos periódicos a serem adicionados.

**MountPoint**

Especifica o diretório onde o sistema de arquivos GFS2 é montado.

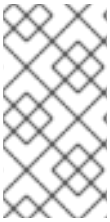
Neste exemplo, uma revista é adicionada ao sistema de arquivos no diretório **/mygfs2**.

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

# CAPÍTULO 4. GESTÃO DE COTAS GFS2

As quotas do sistema de arquivo são usadas para limitar a quantidade de espaço do sistema de arquivo que um usuário ou grupo pode usar. Um usuário ou grupo não tem um limite de cota até que uma seja definida. Quando um sistema de arquivo GFS2 é montado com a opção **quota=on** ou **quota=account**, o GFS2 mantém o controle do espaço usado por cada usuário e grupo, mesmo quando não há limites. O GFS2 atualiza as informações de cota de forma transacional para que as falhas do sistema não exijam a reconstrução do uso da cota.

Para evitar uma diminuição do desempenho, um nó GFS2 sincroniza as atualizações do arquivo de cota apenas periodicamente. A contabilidade de cotas difusas pode permitir que usuários ou grupos excedam ligeiramente o limite estabelecido. Para minimizar isto, o GFS2 reduz dinamicamente o período de sincronização à medida que um limite de cota rígido é aproximado.



NOTA

O GFS2 suporta as instalações padrão de cotas Linux. Para utilizá-lo, você precisará instalar o **quota** RPM. Esta é a forma preferida de administrar cotas no GFS2 e deve ser usada para todas as novas implantações do GFS2 usando cotas. Esta seção documenta a gestão de cotas do GFS2 utilizando estas instalações.

Para mais informações sobre cotas em disco, consulte as páginas **man** dos seguintes comandos:

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

## 4.1. CONFIGURAÇÃO DAS COTAS DE DISCO GFS2

Para implementar cotas em disco, use as seguintes etapas:

1. Estabelecer cotas em modo de aplicação ou contabilidade.
2. Inicializar o arquivo do banco de dados de cotas com as informações atuais de uso do bloco.
3. Atribuir políticas de cotas. (No modo contábil, estas políticas não são aplicadas)

Cada uma dessas etapas é discutida em detalhes nas seções seguintes.

### 4.1.1. Estabelecimento de cotas em modo coercitivo ou contábil

Nos sistemas de arquivos GFS2, as cotas são desabilitadas por padrão. Para ativar as cotas para um sistema de arquivo, monte o sistema de arquivo com a opção **quota=on** especificada.

Para montar um sistema de arquivo com cotas habilitadas, especifique **quota=on** para o argumento **options** ao criar o recurso do sistema de arquivo GFS2 em um cluster. Por exemplo, o seguinte comando especifica que o recurso GFS2 **Filesystem** sendo criado será montado com as cotas ativadas.

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

É possível acompanhar o uso do disco e manter a contabilidade de quotas para cada usuário e grupo sem impor o limite e avisar sobre os valores. Para isso, monte o sistema de arquivos com a opção **quota=account** especificada.

Para montar um sistema de arquivo com cotas desabilitadas, especifique **quota=off** para o argumento **options** ao criar o recurso do sistema de arquivo GFS2 em um cluster.

### 4.1.2. Criação dos arquivos do banco de dados de cotas

Após cada sistema de arquivo habilitado para cotas ser montado, o sistema é capaz de trabalhar com cotas em disco. Entretanto, o sistema de arquivo em si ainda não está pronto para suportar cotas. O próximo passo é executar o comando **quotacheck**.

O comando **quotacheck** examina os sistemas de arquivo habilitados para quotas e constrói uma tabela do uso atual do disco por sistema de arquivo. A tabela é então usada para atualizar a cópia do sistema operacional do uso do disco. Além disso, os arquivos de cota de disco do sistema de arquivos são atualizados.

Para criar os arquivos de cotas no sistema de arquivos, use as opções **-u** e **-g** do comando **quotacheck**; ambas as opções devem ser especificadas para que as cotas de usuários e grupos sejam inicializadas. Por exemplo, se as cotas estiverem habilitadas para o sistema de arquivos /**home**, crie os arquivos no diretório /**home**:

```
quotacheck -ug /home
```

### 4.1.3. Atribuição de cotas por usuário

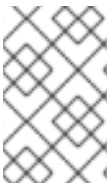
O último passo é atribuir as quotas de disco com o comando **edquota**. Observe que se você montou seu sistema de arquivos em modo contábil (com a opção **quota=account** especificada), as cotas não são aplicadas.

Para configurar a cota para um usuário, como raiz em um prompt de shell, execute o comando:

```
# edquota username
```

Realize esta etapa para cada usuário que necessita de uma cota. Por exemplo, se uma cota for ativada para a partição /**home** (/**dev**/**VolGroup00**/**LogVol02** no exemplo abaixo) e o comando **edquota testuser** for executado, o seguinte é mostrado no editor configurado como padrão para o sistema:

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



NOTA

O editor de texto definido pela variável de ambiente **EDITOR** é utilizado por **edquota**. Para mudar o editor, defina a variável de ambiente **EDITOR** em seu arquivo ~/**.bash_profile** para o caminho completo do editor de sua escolha.

A primeira coluna é o nome do sistema de arquivo que tem uma cota habilitada para ele. A segunda coluna mostra quantos blocos o usuário está utilizando atualmente. As duas colunas seguintes são usadas para definir limites de blocos macios e rígidos para o usuário no sistema de arquivo.

O limite do bloco macio define a quantidade máxima de espaço em disco que pode ser usada.

O limite do bloco rígido é a quantidade máxima absoluta de espaço em disco que um usuário ou grupo pode utilizar. Uma vez atingido este limite, nenhum outro espaço em disco pode ser utilizado.

O sistema de arquivos GFS2 não mantém cotas para inodes, portanto estas colunas não se aplicam aos sistemas de arquivos GFS2 e ficarão em branco.

Se qualquer um dos valores for definido como 0, esse limite não será definido. No editor de texto, altere os limites. Por exemplo:

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

Para verificar se a cota para o usuário foi definida, use o seguinte comando:

```
# quota testuser
```

Você também pode definir cotas a partir da linha de comando com o comando **setquota**. Para obter informações sobre o comando **setquota**, consulte a página de manual **setquota**(8).

### 4.1.4. Atribuição de cotas por grupo

As cotas também podem ser atribuídas por grupo. Observe que se você montou seu sistema de arquivo no modo contábil (com a opção **account=on** especificada), as cotas não são aplicadas.

Para estabelecer uma cota de grupo para o grupo **devel** (o grupo deve existir antes de estabelecer a cota de grupo), use o seguinte comando:

```
# edquota -g devel
```

Este comando exibe a cota existente para o grupo no editor de texto:

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

O sistema de arquivos GFS2 não mantém cotas para inodes, portanto estas colunas não se aplicam aos sistemas de arquivos GFS2 e ficarão em branco. Modifique os limites e, em seguida, salve o arquivo.

Para verificar se a cota do grupo foi definida, use o seguinte comando:

```
$ quota -g devel
```

## 4.2. GESTÃO DE COTAS DE DISCO GFS2

Se as cotas forem implementadas, elas precisam de alguma manutenção, principalmente na forma de vigiar para ver se as cotas são excedidas e certificar-se de que as cotas sejam precisas.

Se os usuários excederem repetidamente suas quotas ou atingirem de forma consistente seus limites de soft, um administrador de sistema tem algumas escolhas a fazer, dependendo do tipo de usuários que são e quanto espaço em disco impacta seu trabalho. O administrador pode ajudar o usuário a determinar

como utilizar menos espaço em disco ou aumentar a cota de disco do usuário.

Você pode criar um relatório de uso do disco executando o utilitário **repquota**. Por exemplo, o comando **repquota /home** produz esta saída:

```
* 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
```

Para visualizar o relatório de uso do disco para todos os sistemas de arquivos habilitados para quotas (opção **-a**), use o comando:

```
# repquota -a
```

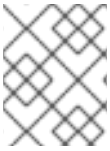
O **--** exibido após cada usuário é uma forma rápida de determinar se os limites do bloco foram excedidos. Se o limite do bloco for excedido, um  aparece no lugar do primeiro **-** na saída. O segundo **-** indica o limite inode, mas os sistemas de arquivo GFS2 não suportam limites inode para que o caractere permaneça como **-**. Os sistemas de arquivo GFS2 não suportam um período de carência, de modo que a coluna **grace** permanecerá em branco.

Note que o comando **repquota** não é suportado sobre o NFS, independentemente do sistema de arquivo subjacente.

## 4.3. MANTENDO AS QUOTAS DE DISCO GFS2 PRECISAS COM O COMANDO DE COTACHECK

Se você ativar cotas em seu sistema de arquivos após um período de tempo em que você tenha sido executado com as cotas desativadas, você deve executar o comando **quotacheck** para criar, verificar e reparar arquivos de cotas. Além disso, você pode querer executar o comando **quotacheck** se achar que seus arquivos de cotas podem não ser precisos, como pode ocorrer quando um sistema de arquivos não é desmontado de forma limpa após uma falha do sistema.

Para mais informações sobre o comando **quotacheck**, consulte a página de manual **quotacheck**.



NOTA

Execute **quotacheck** quando o sistema de arquivo estiver relativamente ocioso em todos os nós, pois a atividade do disco pode afetar os valores das cotas computadas.

## 4.4. SINCRONIZAÇÃO DE COTAS COM O COMANDO DE COTASYNC

O GFS2 armazena em disco todas as informações de cota em seu próprio arquivo interno. Um nó GFS2 não atualiza este arquivo de cota para cada sistema de arquivo escrito; pelo contrário, por padrão, ele atualiza o arquivo de cota uma vez a cada 60 segundos. Isto é necessário para evitar contendas entre os nós que escrevem no arquivo de cota, o que causaria um retardamento no desempenho.

Medida que um usuário ou grupo se aproxima de seu limite de cota, o GFS2 reduz dinamicamente o tempo entre suas atualizações de arquivos de cota para evitar que o limite seja excedido. O período de tempo normal entre as sincronizações de cota é um parâmetro sintonizável, **quota_quantum**. Você pode

mudar isto de seu valor padrão de 60 segundos usando a opção de montagem **quota_quantum=**, como descrito na tabela "GFS2-Specific Mount Options" em Mounting a GFS2 file system que especifica as opções de montagem.

O parâmetro **quota_quantum** deve ser definido em cada nó e cada vez que o sistema de arquivo é montado. As mudanças no parâmetro **quota_quantum** não são persistentes em montagens não montadas. Você pode atualizar o valor **quota_quantum** com o parâmetro **mount -o remount**.

Você pode usar o comando **quotasync** para sincronizar as informações de cota de um nó para o arquivo de cota em disco entre as atualizações automáticas realizadas pelo GFS2. Utilização **Synchronizing Quota Information**

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

**u**

Sincronizar os arquivos de cota de usuários.

**g**

Sincronizar os arquivos de cota do grupo

**a**

Sincronizar todos os sistemas de arquivo que estão atualmente habilitados para cotas e suportar a sincronização. Quando -a estiver ausente, deve ser especificado um ponto de montagem do sistema de arquivo.

**mountpoint**

Especifica o sistema de arquivos GFS2 ao qual as ações se aplicam.

Você pode ajustar o tempo entre as sincronizações especificando uma opção de montagem em **quota-quantum**.

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

**MountPoint**

Especifica o sistema de arquivos GFS2 ao qual as ações se aplicam.

**secs**

Especifica o novo período de tempo entre as sincronizações regulares de arquivos de cota pelo GFS2. Valores menores podem aumentar a contenção e diminuir o desempenho.

O exemplo a seguir sincroniza todas as cotas sujas em cache do nó em que é executado para o arquivo de cotas em disco para o sistema de arquivo **/mnt/mygfs2**.

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

Este exemplo a seguir altera o período de tempo padrão entre atualizações regulares de arquivos de cota para uma hora (3600 segundos) para o sistema de arquivo **/mnt/mygfs2** ao recontar esse sistema de arquivo no volume lógico **/dev/volgroup/logical_volume**.

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

# CAPÍTULO 5. REPARO DO SISTEMA DE ARQUIVOS GFS2

Quando os nós falham com o sistema de arquivo montado, o diário do sistema de arquivo permite uma recuperação rápida. Entretanto, se um dispositivo de armazenamento perder energia ou estiver fisicamente desconectado, pode ocorrer corrupção do sistema de arquivo. (O journaling não pode ser usado para recuperar de falhas no subsistema de armazenamento.) Quando esse tipo de corrupção ocorre, pode-se recuperar o sistema de arquivos GFS2 usando o comando **fsck.gfs2**.

IMPORTANTE

O comando **fsck.gfs2** deve ser executado somente em um sistema de arquivo que não esteja montado a partir de todos os nós. Quando o sistema de arquivo estiver sendo gerenciado como um recurso de cluster de Pacemaker, você pode desativar o recurso do sistema de arquivo, que desmonta o sistema de arquivo. Após executar o comando **fsck.gfs2**, você habilita novamente o recurso do sistema de arquivo. O valor *timeout* especificado com a opção **--wait** do **pcs resource disable** indica um valor em segundos.

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

Para garantir que o comando **fsck.gfs2** não seja executado em um sistema de arquivos GFS2 no momento do boot, você pode definir o parâmetro **run_fsck** do argumento **options** ao criar o recurso do sistema de arquivos GFS2 em um cluster. Especificar **"run_fsck=no"** indicará que você não deve rodar o comando **fsck**.

## 5.1. DETERMINAÇÃO DA MEMÓRIA NECESSÁRIA PARA EXECUTAR O FSCK.GFS2

A execução do comando **fsck.gfs2** pode requerer memória do sistema acima e além da memória utilizada para o sistema operacional e o kernel. Sistemas de arquivos maiores, em particular, podem requerer memória adicional para executar este comando.

A tabela a seguir mostra valores aproximados de memória que podem ser necessários para executar sistemas de arquivo **fsck.gfs2** em sistemas de arquivo GFS2 que são 1TB, 10TB e 100TB de tamanho com um bloco de 4K.

| Tamanho do sistema de arquivo GFS2 | Memória aproximada necessária para rodar **fsck.gfs2** |
|---|---|
| 1 TB | 0.16 GB |
| 10 TB | 1.6 GB |
| 100 TB | 16 GB |

Observe que um bloco de tamanho menor para o sistema de arquivo exigiria uma quantidade maior de memória. Por exemplo, sistemas de arquivo GFS2 com um tamanho de bloco de 1K exigiriam quatro vezes a quantidade de memória indicada nesta tabela.

## 5.2. CONSERTO DE UM SISTEMA DE ARQUIVOS GFS2

O seguinte mostra o formato do comando **fsck.gfs2** para reparar um sistema de arquivos GFS2.

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

**-y**

A bandeira **-y** faz com que todas as perguntas sejam respondidas com **yes**. Com a bandeira **-y** especificada, o comando **fsck.gfs2** não solicita uma resposta antes de fazer mudanças.

**BlockDevice**

Especifica o dispositivo de bloco onde reside o sistema de arquivos GFS2.

Neste exemplo, o sistema de arquivo GFS2 residente no dispositivo de bloco **/dev/testvg/testlv** é reparado. Todas as consultas para reparo são automaticamente respondidas com **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
```

# CAPÍTULO 6. MELHORANDO O DESEMPENHO DO GFS2

Esta seção fornece conselhos para melhorar o GFS2 peformance.

Para recomendações gerais de implantação e atualização dos clusters Red Hat Enterprise Linux usando o Add-On de Alta Disponibilidade e o Red Hat Global File System 2 (GFS2) veja o artigo "Red Hat Enterprise Linux Cluster, High Availability, and GFS Deployment Best Practices" no Portal do Cliente Red Hat em https://access.redhat.com/kb/docs/DOC-40821.

## 6.1. DESFRAGMENTAÇÃO DO SISTEMA DE ARQUIVOS GFS2

Embora não exista uma ferramenta de desfragmentação para o GFS2 no Red Hat Enterprise Linux, você pode desfragmentar arquivos individuais identificando-os com a ferramenta **filefrag**, copiando-os para arquivos temporários e renomeando os arquivos temporários para substituir os originais.

## 6.2. GFS2 BLOQUEIO DE NÓ

A fim de obter o melhor desempenho de um sistema de arquivos GFS2, é importante entender algumas das teorias básicas de seu funcionamento. Um sistema de arquivo de nó único é implementado junto com um cache, cujo objetivo é eliminar a latência dos acessos ao disco quando se utilizam dados freqüentemente solicitados. No Linux, o cache de páginas (e historicamente o cache de buffer) fornece esta função de cache.

Com o GFS2, cada nó tem seu próprio cache de página que pode conter alguma porção dos dados em disco. O GFS2 usa um mecanismo de travamento chamado *glocks* (pronunciado gee-locks) para manter a integridade do cache entre os nós. O subsistema glock fornece uma função de gerenciamento de cache que é implementada usando o *distributed lock manager* (DLM) como a camada de comunicação subjacente.

Os glocks fornecem proteção para o cache por inode, de modo que há um cadeado por inode que é usado para controlar a camada de cache. Se essa glock for concedida em modo compartilhado (modo de bloqueio DLM: PR), então os dados sob essa glock podem ser armazenados em cache em um ou mais nós ao mesmo tempo, de modo que todos os nós possam ter acesso local aos dados.

Se a glock for concedida em modo exclusivo (modo de bloqueio DLM: EX), então somente um único nó pode armazenar os dados sob aquela glock. Este modo é usado por todas as operações que modificam os dados (como a chamada do sistema **write** ).

Se outro nó solicitar uma glock que não possa ser concedida imediatamente, então o DLM envia uma mensagem ao nó ou nós que atualmente seguram as glocks bloqueando o novo pedido para pedir-lhes que soltem suas fechaduras. Deixar cair os glocks pode ser (pelos padrões da maioria das operações do sistema de arquivos) um processo longo. Deixar cair uma glock compartilhada requer apenas que o cache seja invalidado, o que é relativamente rápido e proporcional à quantidade de dados em cache.

Para soltar uma glock exclusiva é necessário um log flush, e escrever de volta quaisquer dados alterados em disco, seguido da invalidação de acordo com a glock compartilhada.

A diferença entre um sistema de arquivo de nó único e o GFS2, então, é que um sistema de arquivo de nó único tem um único cache e o GFS2 tem um cache separado em cada nó. Em ambos os casos, a latência para acessar dados em cache é de uma ordem de magnitude semelhante, mas a latência para acessar dados não em cache é muito maior no GFS2 se outro nó tiver previamente armazenado em cache esses mesmos dados.

Operações como **read** (tamponada), **stat,** e **readdir** requerem apenas uma glock compartilhada. Operações como **write** (tamponada), **mkdir**, **rmdir** e **unlink** requerem uma glock exclusiva. Operações

de leitura/escrita de E/S direta requerem uma glock adiada se nenhuma alocação estiver ocorrendo, ou uma glock exclusiva se a escrita exigir uma alocação (ou seja, extensão do arquivo, ou preenchimento de buracos).

Há duas considerações principais de desempenho que decorrem disso. Em primeiro lugar, as operações somente de leitura paralelizam extremamente bem em um cluster, uma vez que podem funcionar independentemente em cada nó. Segundo, as operações que requerem uma glock exclusiva podem reduzir o desempenho, se houver vários nós disputando o acesso ao(s) mesmo(s) inodo(s). A consideração do conjunto de trabalho em cada nó é, portanto, um fator importante no desempenho do sistema de arquivos GFS2, como, por exemplo, quando se realiza um backup do sistema de arquivos, como descrito em Backing up de um sistema de arquivos GFS2 .

Uma outra conseqüência disto é que recomendamos o uso da opção de montagem **noatime** ou **nodiratime** com GFS2 sempre que possível, com a preferência por **noatime** onde a aplicação permite isto. Isto evita que as leituras necessitem de fechaduras exclusivas para atualizar o timestamp **atime**.

Para usuários preocupados com o conjunto de trabalho ou com a eficiência do cache, o GFS2 fornece ferramentas que permitem monitorar o desempenho de um sistema de arquivos GFS2: Co-piloto de desempenho e pontos de rastreamento GFS2.

NOTA

Devido à forma como o caching do GFS2 é implementado, o melhor desempenho é obtido quando ocorre uma das seguintes ocorrências:

- Um inode é usado de forma somente leitura em todos os nós.
- Um inode é escrito ou modificado a partir de um único nó apenas.

Observe que inserir e remover entradas de um diretório durante a criação e exclusão de arquivos conta como gravação no inode do diretório.

É possível quebrar esta regra desde que ela seja quebrada com relativa freqüência. Ignorar esta regra com demasiada freqüência resultará em uma penalidade severa de desempenho.

Se você **mmap**() um arquivo sobre GFS2 com um mapeamento de leitura/escrita, mas somente leitura a partir dele, isto só conta como uma leitura.

Se você não definir o parâmetro **noatime mount** , então a leitura também resultará em escritas para atualizar os carimbos de tempo do arquivo. Recomendamos que todos os usuários do GFS2 devem montar com **noatime** a menos que tenham uma exigência específica para **atime**.

## 6.3. PROBLEMAS COM O TRAVAMENTO POSIX

Ao utilizar o travamento Posix, você deve levar em conta o seguinte:

- O uso de lotes produzirá um processamento mais rápido do que o uso de travas Posix.
- Programas que utilizam bloqueios Posix no GFS2 devem evitar o uso da função **GETLK** uma vez que, em um ambiente agrupado, a identificação do processo pode ser para um nó diferente no agrupamento.

## 6.4. SINTONIA DE DESEMPENHO COM GFS2

Geralmente é possível alterar a forma como uma aplicação problemática armazena seus dados a fim de obter uma vantagem considerável de desempenho.

Um exemplo típico de uma aplicação problemática é um servidor de e-mail. Estes são freqüentemente dispostos com um diretório spool contendo arquivos para cada usuário (**mbox**), ou com um diretório para cada usuário contendo um arquivo para cada mensagem (**maildir**). Quando os pedidos chegam por IMAP, a disposição ideal é dar a cada usuário uma afinidade com um nó em particular. Dessa forma, suas solicitações para visualizar e excluir mensagens de e-mail tenderão a ser servidas a partir do cache daquele nó. Obviamente, se esse nó falhar, então a sessão poderá ser reiniciada em um nó diferente.

Quando o correio chega por meio de SMTP, então novamente os nós individuais podem ser configurados de forma a passar um determinado correio do usuário para um determinado nó por padrão. Se o nó padrão não estiver ativo, então a mensagem pode ser salva diretamente no spool de correio do usuário pelo nó de recebimento. Mais uma vez, este projeto visa manter conjuntos particulares de arquivos em cache em apenas um nó no caso normal, mas para permitir acesso direto no caso de falha do nó.

Esta configuração permite o melhor uso do cache de páginas do GFS2 e também torna as falhas transparentes para a aplicação, seja **imap** ou **smtp**.

O Backup é muitas vezes outra área complicada. Novamente, se for possível, é muito preferível fazer o backup do conjunto de trabalho de cada nó diretamente do nó que está armazenando aquele conjunto particular de nós. Se você tem um script de backup que roda em um ponto regular no tempo, e isso parece coincidir com um pico no tempo de resposta de uma aplicação rodando no GFS2, então há uma boa chance de que o cluster possa não estar fazendo o uso mais eficiente do cache de páginas.

Obviamente, se você estiver na posição de poder interromper a aplicação para realizar um backup, então isto não será um problema. Por outro lado, se um backup for executado a partir de apenas um nó, então, depois de ter completado uma grande parte do sistema de arquivos será colocado em cache naquele nó, com uma penalidade de desempenho para acessos subseqüentes a partir de outros nós. Isto pode ser mitigado, até certo ponto, deixando cair o cache de páginas VFS no nó de backup depois que o backup for completado com o seguinte comando:

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

No entanto, esta não é uma solução tão boa quanto o cuidado de garantir que o conjunto de trabalho em cada nó seja compartilhado, em sua maioria somente de leitura, ou acessado em grande parte a partir de um único nó.

## 6.5. SOLUÇÃO DE PROBLEMAS DE DESEMPENHO DO GFS2 COM O LIXÃO DE FECHADURA GFS2

Se seu desempenho de cluster estiver sofrendo devido ao uso ineficiente do cache GFS2, você poderá ver grandes e crescentes tempos de espera de E/S. Você pode fazer uso das informações de lock dump do GFS2 para determinar a causa do problema.

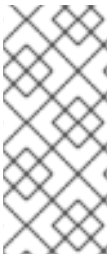
Esta seção fornece uma visão geral do lixão de eclusas GFS2.

As informações sobre o lixão GFS2 podem ser coletadas no arquivo **debugfs** que pode ser encontrado no seguinte caminho, assumindo que **debugfs** esteja montado em **/sys/kernel/debug/**:

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

O conteúdo do arquivo é uma série de linhas. Cada linha começando com G: representa uma glock, e as seguintes linhas, recuadas por um único espaço, representam um item de informação relativo à glock imediatamente antes delas no arquivo.

A melhor maneira de usar o arquivo **debugfs** é usar o comando **cat** para obter uma cópia do conteúdo completo do arquivo (pode levar muito tempo se você tiver uma grande quantidade de RAM e muitos inodes em cache) enquanto a aplicação está enfrentando problemas, e então olhar os dados resultantes em uma data posterior.



NOTA

Pode ser útil fazer duas cópias do arquivo **debugfs**, uma poucos segundos ou até mesmo um minuto ou dois depois da outra. Comparando as informações do titular nos dois traços relativos ao mesmo número de glock, você pode dizer se a carga de trabalho está progredindo (é apenas lenta) ou se ficou presa (o que é sempre um bug e deve ser reportado ao suporte da Red Hat imediatamente).

As linhas no arquivo **debugfs** que começam com H: (titulares) representam pedidos de fechaduras concedidas ou à espera de serem concedidas. O campo das bandeiras na linha f: mostra qual: A bandeira "W" se refere a um pedido de espera, a bandeira "H" se refere a um pedido deferido. Os glocks que têm um grande número de pedidos de espera são provavelmente aqueles que estão passando por uma disputa particular.

Tabela 6.1, “Bandeiras Glock” mostra os significados das diferentes bandeiras porta-bandeiras e Tabela 6.2, “Bandeiras porta-bandeiras Glock” mostra os significados das diferentes bandeiras porta-bandeiras.

**Tabela 6.1. Bandeiras Glock**

| Bandeira | Nome | Significado |
|---|---|---|
| b | Bloqueio | Válido quando a bandeira bloqueada é colocada, e indica que a operação que foi solicitada ao DLM pode bloquear. Esta bandeira é liberada para as operações de despromoção e para os bloqueios "try". O objetivo desta bandeira é permitir a coleta de estatísticas do tempo de resposta do DLM independente do tempo que os outros nós levam para destravar as fechaduras. |
| d | Pendente de demote | Um pedido de demote (remoto) diferido |
| D | Demote | Um pedido de demote (local ou remoto) |
| f | Fluxo de toras | O tronco precisa ser comprometido antes de liberar esta glóculo |

| Bandeira | Nome | Significado |
|---|---|---|
| F | Congelados | Respostas de nós remotos ignorados - a recuperação está em andamento. Esta bandeira não está relacionada ao congelamento do sistema de arquivos, que usa um mecanismo diferente, mas é usada apenas na recuperação. |
| i | Invalidar em andamento | No processo de invalidar as páginas sob esta glock |
| I | Inicial | Definido quando a fechadura DLM é associada a esta glock |
| l | Fechado | A glóculo está em processo de mudança de estado |
| L | LRU | Definido quando a glock está na lista da LRU |
| o | Objeto | Definido quando a glock está associada a um objeto (ou seja, um inode para glocks tipo 2 e um grupo de recursos para glocks tipo 3) |
| p | Demonstração em andamento | A glock está no processo de responder a um pedido de demote |
| q | Enfileirado | Definido quando um suporte é enfileirado e limpo quando a glande é mantida, mas não há nenhum suporte restante. Usado como parte do algoritmo, o algoritmo calcula o tempo mínimo de retenção para uma bata. |
| r | Resposta pendente | A resposta recebida do nó remoto está aguardando processamento |
| y | Sujo | Os dados precisam ser lavados em disco antes de soltar esta glock |

Tabela 6.2. Bandeiras porta-bandeiras Glock

| Bandeira | Nome | Significado |
|---|---|---|
| a | Async | Não espere pelo resultado da glock (pesquisará o resultado mais tarde) |
| A | Qualquer | Qualquer modo de bloqueio compatível é aceitável |
| c | Sem cache | Quando desbloqueado, destravar imediatamente a fechadura DLM |
| e | Não expira | Ignorar pedidos subsequentes de cancelamento de bloqueio |
| E | exato | Deve ter o modo de bloqueio exato |
| F | Primeiro | Definido quando o titular é o primeiro a ser concedido para esta fechadura |
| H | Titular | Indica que o bloqueio solicitado é concedido |
| p | Prioridade | Titular da fila de espera à frente da fila |
| t | Tente | Uma fechadura "try |
| T | Experimente 1CB | Uma fechadura "try" que envia uma chamada de retorno |
| W | Aguarde | Definido enquanto se aguarda a solicitação para completar |

Tendo identificado uma glock que está causando um problema, o próximo passo é descobrir com qual inode ela se relaciona. O número da glóculo (n: na linha G:) indica isto. É do formulário *type*/*number* e se *type* é 2, então a glock é um inode glock e o *number* é um inode number. Para rastrear o inode, você pode então executar **find -inum *number*** onde *number* é o número do inode convertido do formato hexadecimal no arquivo de glocks em decimal.

**ATENÇÃO**

Se você executar o comando **find** em um sistema de arquivo quando ele estiver passando por uma contenção de travas, é provável que você piore o problema. É uma boa idéia parar a aplicação antes de executar o comando **find** quando você estiver procurando por inodes contendidos.

Tabela 6.3, "Tipos de Glock" mostra os significados dos diferentes tipos de glock.

**Tabela 6.3. Tipos de Glock**

| Número do tipo | Tipo de fechadura | Use |
|---|---|---|
| 1 | Trans | Trava da transação |
| 2 | Inode | Inode metadados e dados |
| 3 | Rgrp | Metadados dos grupos de recursos |
| 4 | Meta | O superbloco |
| 5 | Iopen | A última detecção mais próxima do inode |
| 6 | Rebanho | **flock**(2) syscall |
| 8 | Quota | Operações de cotas |
| 9 | Jornal | Jornal mutex |

Se a glock que foi identificada era de um tipo diferente, então é mais provável que seja do tipo 3: (grupo de recursos). Se você vir um número significativo de processos esperando por outros tipos de glock sob cargas normais, informe isso ao suporte da Red Hat.

Se você vir uma série de pedidos em fila de espera em um bloqueio de grupo de recursos, pode haver uma série de razões para isso. Uma delas é que há um grande número de nós em comparação com o número de grupos de recursos no sistema de arquivos. Outra é que o sistema de arquivos pode estar muito próximo de estar cheio (exigindo, em média, buscas mais longas por blocos livres). A situação em ambos os casos pode ser melhorada adicionando mais armazenamento e usando o comando **gfs2_grow** para expandir o sistema de arquivo.

## 6.6. PERMITINDO O DIÁRIO DE DADOS

Normalmente, o GFS2 escreve apenas metadados para sua revista. O conteúdo do arquivo é posteriormente escrito em disco pela sincronização periódica do kernel que descarrega os buffers do sistema de arquivos. Uma chamada ao arquivo **fsync()** faz com que os dados do arquivo sejam

imediatamente gravados em disco. A chamada retorna quando o disco informa que todos os dados estão gravados com segurança.

O diário de dados pode resultar em uma redução do tempo **fsync()** para arquivos muito pequenos porque os dados do arquivo são escritos no diário, além dos metadados. Esta vantagem se reduz rapidamente à medida que o tamanho do arquivo aumenta. A escrita em arquivos médios e grandes será muito mais lenta com o diário de dados ligado.

As aplicações que dependem do **fsync()** para sincronizar dados de arquivos podem ver um melhor desempenho ao usar o data journaling. O diário de dados pode ser ativado automaticamente para qualquer arquivo GFS2 criado em um diretório sinalizado (e todos os seus subdiretórios). Os arquivos existentes com comprimento zero também podem ter o diário de dados ligado ou desligado.

A ativação do diário de dados em um diretório define o diretório como "herdar jdata", o que indica que todos os arquivos e diretórios criados subseqüentemente nesse diretório são diários. Você pode ativar e desativar o diário de dados em um arquivo com o comando **chattr**.

Os comandos a seguir permitem o diário de dados no arquivo **/mnt/gfs2/gfs2_dir/newfile** e, em seguida, verificar se a bandeira foi colocada corretamente.

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

Os seguintes comandos desativam o diário de dados no arquivo **/mnt/gfs2/gfs2_dir/newfile** e depois verificam se a bandeira foi colocada corretamente.

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

Você também pode usar o comando **chattr** para colocar a bandeira **j** em um diretório. Quando você configura esta bandeira para um diretório, todos os arquivos e diretórios criados subseqüentemente nesse diretório são registrados no diário. O conjunto de comandos a seguir coloca a bandeira **j** no diretório **gfs2_dir** e, em seguida, verifica se a bandeira foi colocada corretamente. Depois disso, os comandos criam um novo arquivo chamado **newfile** no diretório **/mnt/gfs2/gfs2_dir** e então verificam se o sinalizador **j** foi definido para o arquivo. Como a bandeira **j** está configurada para o diretório, então **newfile** também deve ter o journaling habilitado.

```
# 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
```

# CAPÍTULO 7. DIAGNOSTICAR E CORRIGIR PROBLEMAS COM OS SISTEMAS DE ARQUIVO GFS2

Esta seção fornece informações sobre algumas questões comuns do GFS2 e como tratá-las.

## 7.1. SISTEMA DE ARQUIVOS GFS2 INDISPONÍVEL PARA UM NÓ (A FUNÇÃO DE RETIRADA GFS2)

A função GFS2 *withdraw* é uma característica de integridade de dados do sistema de arquivos GFS2 que evita possíveis danos ao sistema de arquivos devido a hardware ou software do kernel defeituoso. Se o módulo de kernel GFS2 detectar uma inconsistência ao usar um sistema de arquivo GFS2 em um determinado nó de cluster, ele se retira do sistema de arquivo, deixando-o indisponível para aquele nó até que seja desmontado e remontado (ou a máquina que detecta o problema seja reinicializada). Todos os outros sistemas de arquivo GFS2 montados permanecem totalmente funcionais naquele nó. (A função de retirada do GFS2 é menos severa do que um pânico no núcleo, o que faz com que o nó seja cercado)

As principais categorias de inconsistência que podem causar uma retirada do GFS2 são as seguintes:

- Erro de consistência do inodo
- Erro de consistência do grupo de recursos
- Erro de consistência do diário
- Erro de consistência de metadados de número mágico
- Erro de consistência do tipo Metadata

Um exemplo de uma inconsistência que causaria uma retirada do GFS2 é uma contagem de blocos incorreta para o inode de um arquivo. Quando o GFS2 apaga um arquivo, ele remove sistematicamente todos os blocos de dados e metadados referenciados por aquele arquivo. Quando feito, ele verifica a contagem de blocos do inode. Se a contagem de blocos não for 1 (significando que tudo o que resta é o próprio inode do disco), isso indica uma inconsistência do sistema de arquivo, uma vez que a contagem de blocos do inode não correspondeu aos blocos reais usados para o arquivo.

Em muitos casos, o problema pode ter sido causado por hardware defeituoso (memória defeituosa, placa-mãe, HBA, drives de disco, cabos, e assim por diante). Pode também ter sido causado por um bug no kernel (outro módulo do kernel sobregravando acidentalmente a memória do GFS2), ou por danos reais ao sistema de arquivos (causados por um bug GFS2).

Na maioria dos casos, a melhor maneira de recuperar de um sistema de arquivo GFS2 retirado é reinicializar ou cercar o nó. O sistema de arquivo GFS2 retirado lhe dará a oportunidade de realocar os serviços para outro nó no cluster. Após a realocação dos serviços, você pode reinicializar o nó ou forçar uma cerca com este comando.

```
# pcs stonith fence node
```

**ATENÇÃO**

Não tente desmontar e remontar o sistema de arquivo manualmente com os comandos **umount** e **mount**. Você deve usar o comando **pcs**, caso contrário o Pacemaker detectará que o serviço do sistema de arquivo desapareceu e cercará o nó.

O problema de consistência que causou a retirada pode impossibilitar a interrupção do serviço do sistema de arquivos, pois pode causar a suspensão do sistema.

Se o problema persistir após uma montagem posterior, você deve parar o serviço de sistema de arquivo para desmontar o sistema de arquivo de todos os nós no cluster, então execute uma verificação do sistema de arquivo com o comando fsck.gfs2 antes de reiniciar o serviço com o seguinte procedimento.

1. Reinicializar o nó afetado.
2. Desabilite o serviço de sistema de arquivo não clone no Pacemaker para desmontar o sistema de arquivo de cada nó do cluster.

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

3. A partir de um nó do cluster, execute o comando **fsck.gfs2** no dispositivo do sistema de arquivos para verificar e reparar qualquer dano ao sistema de arquivos.

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

4. Remonte o sistema de arquivos GFS2 de todos os nós, ativando novamente o serviço de sistema de arquivos:

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

Você pode anular a função de retirada do GFS2 montando o sistema de arquivo com a opção **-o errors=panic** especificada no serviço de sistema de arquivo.

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

Quando esta opção é especificada, quaisquer erros que normalmente causariam a retirada do sistema forçam um pânico no núcleo. Isto interrompe as comunicações do nó, o que faz com que o nó seja cercado. Isto é especialmente útil para clusters que são deixados sem vigilância por longos períodos de tempo sem monitoramento ou intervenção.

Internamente, a função de retirada do GFS2 funciona desligando o protocolo de travamento para garantir que todas as demais operações do sistema de arquivos resultem em erros de E/S. Como resultado, quando a retirada ocorre, é normal ver uma série de erros de E/S do dispositivo de mapeamento relatados nos logs do sistema.

## 7.2. O SISTEMA DE ARQUIVO GFS2 FICA PENDURADO E REQUER A REINICIALIZAÇÃO DE UM NÓ

Se seu sistema de arquivos GFS2 estiver pendurado e não retornar comandos executados contra ele, mas reiniciar um nó específico retorna o sistema ao normal, isto pode ser indicativo de um problema de travamento ou bug. Caso isso ocorra, colete os dados GFS2 durante uma dessas ocorrências e abra um ticket de suporte com o Red Hat Support, conforme descrito em Coleta de dados GFS2 para solução de problemas.

## 7.3. O SISTEMA DE ARQUIVO GFS2 FICA PENDURADO E REQUER A REINICIALIZAÇÃO DE TODOS OS NÓS

Se seu sistema de arquivos GFS2 estiver pendurado e não retornar comandos executados contra ele, exigindo que você reinicie todos os nós do cluster antes de usá-lo, verifique as seguintes questões.

- Você pode ter tido uma cerca fracassada. Os sistemas de arquivos GFS2 congelarão para garantir a integridade dos dados no caso de uma cerca falhada. Verifique os registros de mensagens para ver se há alguma cerca falhada no momento do enforcamento. Certifique-se de que as cercas estejam configuradas corretamente.

- O sistema de arquivos GFS2 pode ter sido retirado. Verifique através dos registros de mensagens a palavra **withdraw** e verifique se há mensagens e traços de chamadas do GFS2 indicando que o sistema de arquivo foi retirado. Uma retirada é indicativa de corrupção do sistema de arquivos, falha no armazenamento ou um bug. No primeiro momento em que for conveniente desmontar o sistema de arquivo, você deve realizar o seguinte procedimento:

  a. Reiniciar o nó em que ocorreu a retirada.

  ```
  # /sbin/reboot
  ```

  b. Parar o recurso do sistema de arquivos para desmontar o sistema de arquivos GFS2 em todos os nós.

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

  c. Capture os metadados com o comando **gfs2_edit savemeta…**. Você deve garantir que haja espaço suficiente para o arquivo, que em alguns casos pode ser grande. Neste exemplo, os metadados são salvos em um arquivo no diretório **/root**.

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

  d. Atualize o pacote **gfs2-utils**.

  ```
  # sudo yum update gfs2-utils
  ```

  e. Em um nó, execute o comando **fsck.gfs2** no sistema de arquivos para garantir a integridade do sistema de arquivos e reparar qualquer dano.

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

  f. Após o comando **fsck.gfs2** ter sido concluído, reative o recurso do sistema de arquivos para devolvê-lo ao serviço:

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

g. Abra um ticket de suporte com o Red Hat Support. Informe-os que você experimentou uma retirada do GFS2 e forneça os logs e as informações de depuração geradas pelos comandos **sosreports** e **gfs2_edit savemeta**.
Em alguns casos de retirada de um GFS2, comandos podem ser pendurados que estão tentando acessar o sistema de arquivos ou seu dispositivo de bloco. Nesses casos, é necessário um reinício rígido para reiniciar o cluster.

Para informações sobre a função de retirada GFS2, consulte o sistema de arquivos GFS2 indisponível para um nó (a função de retirada GFS2).

- Este erro pode ser indicativo de um problema de travamento ou bug. Reúna dados durante uma dessas ocorrências e abra um ticket de suporte com o Red Hat Support, como descrito em Gathering GFS2 data for troubleshooting.

## 7.4. O SISTEMA DE ARQUIVO GFS2 NÃO É MONTADO EM UM NOVO NÓ DE CLUSTER

Se você adicionar um novo nó a um cluster e descobrir que não pode montar seu sistema de arquivos GFS2 naquele nó, você poderá ter menos periódicos no sistema de arquivos GFS2 do que nós que tentam acessar o sistema de arquivos GFS2. Você deve ter um periódico por host GFS2 no qual pretende montar o sistema de arquivo (com exceção dos sistemas de arquivo GFS2 montados com o conjunto de opções de montagem **spectator**, uma vez que estes não requerem um periódico). Você pode adicionar periódicos a um sistema de arquivo GFS2 com o comando **gfs2_jadd**. Adicionando periódicos a um sistema de arquivo GFS2.

## 7.5. ESPAÇO INDICADO COMO UTILIZADO EM SISTEMA DE ARQUIVO VAZIO

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. COLETA DE DADOS GFS2 PARA SOLUÇÃO DE PROBLEMAS

Se seu sistema de arquivos GFS2 estiver pendurado e não retornar comandos executados contra ele e você descobrir que precisa abrir um ticket com o Suporte da Red Hat, você deve primeiro reunir os seguintes dados:

- O depósito GFS2 para o sistema de arquivo em cada nó:

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

- O depósito de trava DLM para o sistema de arquivo em cada nó: Você pode obter estas informações com o **dlm_tool**:

```
dlm_tool lockdebug -sv lsname.
```

Neste comando, *lsname* é o nome de lockspace usado pela DLM para o sistema de arquivos em questão. Você pode encontrar este valor na saída do comando **group_tool**.

- A saída do comando **sysrq -t**.
- O conteúdo do arquivo /**var**/**log**/**messages**.

Uma vez coletados esses dados, você pode abrir um ticket com o Suporte da Red Hat e fornecer os dados coletados.

# CAPÍTULO 8. DEPURAÇÃO DE SISTEMAS DE ARQUIVO GFS2 COM TRACEPOINTS GFS2 E O ARQUIVO DE DEPURAÇÃO DE GLOCKS GFS2

Esta seção descreve tanto a interface da glock **debugfs** quanto os pontos de rastreamento GFS2. Ela se destina a usuários avançados que estão familiarizados com os internos do sistema de arquivos que gostariam de aprender mais sobre o design do GFS2 e como depurar questões específicas do GFS2.

## 8.1. GFS2 TIPOS DE TRACEPOINT

Existem atualmente três tipos de tracepoints GFS2: *glock* (pronuncia-se "gee-lock") tracepoints, *bmap* tracepoints e *log* tracepoints. Estes podem ser usados para monitorar um sistema de arquivo GFS2 em execução e dar informações adicionais àquelas que podem ser obtidas com as opções de depuração suportadas em versões anteriores do Red Hat Enterprise Linux. Os tracepoints são particularmente úteis quando um problema, tal como um pendrive ou um problema de desempenho, é reproduzível e, portanto, a saída de tracepoints pode ser obtida durante a operação problemática. No GFS2, os glocks são o principal mecanismo de controle do cache e são a chave para entender o desempenho do núcleo do GFS2. O bmap (mapa de blocos) tracepoints pode ser usado para monitorar alocações de blocos e mapeamento de blocos (busca de blocos já alocados na árvore de metadados em disco) à medida que eles acontecem e verificar quaisquer questões relacionadas à localidade de acesso. Os log tracepoints acompanham os dados que estão sendo escritos e liberados da revista e podem fornecer informações úteis sobre essa parte do GFS2.

Os traços são projetados para serem o mais genéricos possível. Isto deve significar que não será necessário alterar a API durante o curso do Red Hat Enterprise Linux 8. Por outro lado, os usuários desta interface devem estar cientes de que esta é uma interface de depuração e não faz parte do conjunto normal da API do Red Hat Enterprise Linux 8, e como tal a Red Hat não dá garantias de que não ocorram mudanças na interface de tracepoints GFS2.

Os tracepoints são uma característica genérica do Red Hat Enterprise Linux e seu escopo vai muito além do GFS2. Em particular, eles são usados para implementar a infra-estrutura **blktrace** e o **blktrace** tracepoints podem ser usados em combinação com os do GFS2 para obter uma imagem mais completa do desempenho do sistema. Devido ao nível em que os tracepoints operam, eles podem produzir grandes volumes de dados em um período de tempo muito curto. Eles são projetados para colocar uma carga mínima no sistema quando são habilitados, mas é inevitável que eles tenham algum efeito. A filtragem de eventos por diversos meios pode ajudar a reduzir o volume de dados e ajudar a concentrar-se na obtenção apenas das informações úteis para a compreensão de qualquer situação particular.

## 8.2. TRACEPOINTS

Os tracepoints podem ser encontrados sob o diretório **/sys/kernel/debug/tracing/** assumindo que **debugfs** esteja montado no local padrão no diretório **/sys/kernel/debug**. O subdiretório **events** contém todos os eventos de rastreamento que podem ser especificados e, desde que o módulo **gfs2** seja carregado, haverá um subdiretório **gfs2** contendo outros subdiretórios, um para cada evento GFS2. O conteúdo do diretório **/sys/kernel/debug/tracing/events/gfs2** deve se parecer mais ou menos com o seguinte:

```
[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
```

Para habilitar todos os tracepoints GFS2, digite o seguinte comando:

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

Para permitir um traço específico, há um arquivo **enable** em cada um dos subdiretórios de eventos individuais. O mesmo se aplica ao arquivo **filter** que pode ser usado para definir um filtro de eventos para cada evento ou conjunto de eventos. O significado dos eventos individuais é explicado com mais detalhes abaixo.

A saída dos tracepoints está disponível em ASCII ou em formato binário. Este apêndice não cobre atualmente a interface binária. A interface ASCII está disponível de duas maneiras. Para listar o conteúdo atual do buffer de anéis, você pode digitar o seguinte comando:

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

Esta interface é útil nos casos em que você está usando um processo de longa duração por um determinado período de tempo e, após algum evento, quer olhar para trás para as últimas informações capturadas no buffer. Uma interface alternativa, **/sys/kernel/debug/tracing/trace_pipe**, pode ser usada quando toda a saída for necessária. Os eventos são lidos a partir deste arquivo à medida que ocorrem; não há informações históricas disponíveis através desta interface. O formato da saída é o mesmo de ambas as interfaces e é descrito para cada um dos eventos GFS2 nas últimas seções deste apêndice.

Um utilitário chamado **trace-cmd** está disponível para a leitura de dados de tracepoint. Para maiores informações sobre este utilitário, veja o link em Seção 8.10, “Referências”. O utilitário **trace-cmd** pode ser usado de forma similar ao utilitário **strace**, por exemplo, para executar um comando durante a coleta de dados de rastreamento de várias fontes.

## 8.3. GLOCKS

Para entender o GFS2, o conceito mais importante a entender, e o que o diferencia de outros sistemas de arquivo, é o conceito de glocks. Em termos de código fonte, um glock é uma estrutura de dados que reúne o DLM e o caching em uma única máquina de estado. Cada glock tem uma relação 1:1 com uma única fechadura DLM, e fornece cache para esse estado de fechamento, de modo que as operações repetitivas realizadas a partir de um único nó do sistema de arquivo não tenham que chamar repetidamente o DLM, e assim ajudam a evitar tráfego desnecessário na rede. Há duas grandes categorias de glocks, aqueles que armazenam metadados e aqueles que não o fazem. Os glocks de inode e o grupo de recursos fazem ambos os metadados de cache, outros tipos de glocks não fazem o cache de metadados. O inode glock também está envolvido no cache de dados além dos metadados e tem a lógica mais complexa de todos os glocks.

**Tabela 8.1. Modos Glock e DLM Lock**

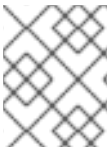
| Modo Glock | Modo de bloqueio DLM | Notas |
|---|---|---|
| ONU | IV/NL | Desbloqueado (sem fechadura DLM associada à glock ou à NL, dependendo da bandeira I) |
| SH | PR | Fechadura compartilhada (leitura protegida) |
| EX | EX | Fechadura exclusiva |

| Modo Glock | Modo de bloqueio DLM | Notas |
|---|---|---|
| DF | CW | Diferido (escrita simultânea) utilizado para E/S direta e congelamento do sistema de arquivo |

As glocks permanecem na memória até que sejam desbloqueadas (a pedido de outro nó ou a pedido do VM) e não haja usuários locais. Nesse momento, elas são removidas da mesa de hash da glock e liberadas. Quando uma glock é criada, o bloqueio DLM não é associado à glock imediatamente. A trava DLM torna-se associada à glock na primeira solicitação à DLM, e se esta solicitação for bem sucedida, então a bandeira "I" (inicial) será colocada na glock. Tabela 8.4, “Bandeiras Glock” mostra os significados das diferentes bandeiras da glock. Uma vez que a DLM tenha sido associada à glock, a trava DLM permanecerá sempre pelo menos no modo NL (Null) até que a glock seja liberada. Uma despromoção da trava DLM de NL para desbloqueada é sempre a última operação na vida de uma bata.

Cada glock pode ter um número de "suportes" associados a ela, cada um dos quais representa um pedido de bloqueio das camadas superiores. Chamadas de sistema relativas aos suportes de fila e dequeue GFS2 da glock para proteger a seção crítica do código.

A máquina de estado de glock é baseada em uma fila de trabalho. Por razões de desempenho, seria preferível a tasklets; entretanto, na implementação atual, precisamos apresentar E/S daquele contexto que proíbe seu uso.



**NOTA**

As filas de trabalho têm seus próprios tracepoints que podem ser usados em combinação com os tracepoints GFS2.

Tabela 8.2, “Modos e tipos de dados da Glock” mostra que estado pode estar em cache sob cada um dos modos de glock e se esse estado em cache pode estar sujo. Isto se aplica tanto aos bloqueios inode quanto aos do grupo de recursos, embora não haja nenhum componente de dados para os bloqueios do grupo de recursos, apenas metadados.

**Tabela 8.2. Modos e tipos de dados da Glock**

| Modo Glock | Dados do Cache | Metadados de Cache | Dados sujos | Metadados sujos |
|---|---|---|---|---|
| ONU | Não | Não | Não | Não |
| SH | Sim | Sim | Não | Não |
| DF | Não | Sim | Não | Não |
| EX | Sim | Sim | Sim | Sim |

## 8.4. A INTERFACE GLOCK DEBUGFS

A interface **debugfs** da glock permite a visualização do estado interno das glocks e dos suportes e

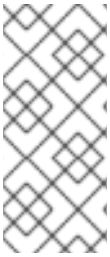
também inclui alguns detalhes resumidos dos objetos que estão sendo trancados em alguns casos. Cada linha do arquivo ou começa com G: sem recuo (que se refere à própria glock) ou começa com uma letra diferente, recuada com um único espaço, e refere-se às estruturas associadas à glock imediatamente acima dela no arquivo (H: é um suporte, I: um inode, e R: um grupo de recursos) . Aqui está um exemplo de como poderia ser o conteúdo deste arquivo:

```
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]
```

O exemplo acima é uma série de trechos (de um arquivo de aproximadamente 18MB) gerados pelo comando **cat /sys/kernel/debug/gfs2/unity:myfs/glocks >my.lock** durante uma execução do benchmark do carimbo do correio em um único nó do sistema de arquivos GFS2. Os glocks da figura foram selecionados a fim de mostrar algumas das características mais interessantes das lixeiras de glock.

Os estados de glock são EX (exclusivo), DF (diferido), SH (compartilhado) ou UN (desbloqueado). Estes estados correspondem diretamente aos modos de bloqueio DLM, exceto ONU, que pode representar ou o estado de bloqueio DLM nulo, ou que o GFS2 não possui um bloqueio DLM (dependendo da bandeira I, conforme explicado acima). O s: campo da glock indica o estado atual da fechadura e o mesmo campo no suporte indica o modo solicitado. Se a fechadura for concedida, o detentor terá o bit H ajustado em suas bandeiras (f: campo). Caso contrário, ele terá o bit W de espera ajustado.

O n: campo (número) indica o número associado a cada item. Para glocks, este é o número do tipo seguido pelo número da glock, de modo que no exemplo acima, a primeira glock é n:5/75320; o que indica uma glock **iopen** que se relaciona ao inode 75320. No caso das glocos inode e **iopen**, o número da glock é sempre idêntico ao número do bloco de disco do inode.



NOTA

Os números de glock (n: campo) no arquivo de glocks debugfs estão em hexadecimal, enquanto que a saída dos tracepoints os lista em decimal. Isto por razões históricas; os números de glock sempre foram escritos em hexadecimal, mas o decimal foi escolhido para os tracepoints de modo que os números pudessem ser facilmente comparados com os outros tracepoints de saída (de **blktrace** por exemplo) e com a saída de **stat**(1).

A listagem completa de todas as bandeiras, tanto para o titular como para a glande, encontra-se em Tabela 8.4, “Bandeiras Glock” e Tabela 8.5, “Bandeiras porta-bandeiras Glock” . O conteúdo dos blocos de valores de bloqueio não está atualmente disponível através da interface da glock **debugfs**.

Tabela 8.3, "Tipos de Glock" mostra os significados dos diferentes tipos de glock.

**Tabela 8.3. Tipos de Glock**

| Número do tipo | Tipo de fechadura | Use |
|---|---|---|
| 1 | trans | Trava da transação |
| 2 | inode | Inode metadados e dados |
| 3 | rgrp | Metadados dos grupos de recursos |
| 4 | meta | O superbloco |
| 5 | iopen | A última detecção mais próxima do inode |
| 6 | bando | **flock**(2) syscall |
| 8 | quota | Operações de cotas |
| 9 | revista | Jornal mutex |

Uma das bandeiras glock mais importantes é a bandeira l (bloqueada). Esta é a bit lock que é usada para arbitrar o acesso ao estado de glock quando uma mudança de estado deve ser realizada. Ela é definida quando a máquina de estado está prestes a enviar um pedido de bloqueio remoto através do DLM, e só é liberada quando a operação completa tiver sido realizada. Às vezes isto pode significar que mais de uma solicitação de bloqueio terá sido enviada, com várias invalidações ocorrendo entre as vezes.

Tabela 8.4, "Bandeiras Glock" mostra os significados das diferentes bandeiras de glock.

**Tabela 8.4. Bandeiras Glock**

| Bandeira | Nome | Significado |
|---|---|---|
| d | Pendente de demote | Um pedido de demote (remoto) diferido |
| D | Demote | Um pedido de demote (local ou remoto) |
| f | Fluxo de toras | O tronco precisa ser comprometido antes de liberar esta glóculo |
| F | Congelados | Respostas de nós remotos ignorados - a recuperação está em andamento. |

| Bandeira | Nome | Significado |
|---|---|---|
| i | Invalidar em andamento | No processo de invalidar as páginas sob esta glock |
| I | Inicial | Definido quando a fechadura DLM é associada a esta glock |
| l | Fechado | A glóculo está em processo de mudança de estado |
| L | LRU | Definido quando a glock está na lista da LRU` |
| o | Objeto | Definido quando a glock está associada a um objeto (ou seja, um inode para glocks tipo 2 e um grupo de recursos para glocks tipo 3) |
| p | Demonstração em andamento | A glock está no processo de responder a um pedido de demote |
| q | Enfileirado | Definido quando um suporte é enfileirado e limpo quando a glande é mantida, mas não há nenhum suporte restante. Usado como parte do algoritmo, o algoritmo calcula o tempo mínimo de retenção para uma bata. |
| r | Resposta pendente | A resposta recebida do nó remoto está aguardando processamento |
| y | Sujo | Os dados precisam ser lavados em disco antes de soltar esta glock |

Quando uma chamada remota é recebida de um nó que deseja obter uma fechadura em um modo que entre em conflito com o que está sendo mantido no nó local, então uma ou outra das duas bandeiras D (demote) ou d (demote pending) é definida. A fim de evitar condições de fome quando há contendas em uma determinada eclusa, a cada eclusa é atribuído um tempo mínimo de retenção. Um nó que ainda não teve a eclusa pelo tempo mínimo de retenção é autorizado a reter essa eclusa até que o intervalo de tempo tenha expirado.

Se o intervalo de tempo tiver expirado, então a bandeira D (demotada) será colocada e o estado requerido será registrado. Nesse caso, na próxima vez que não houver fechaduras concedidas na fila de titulares, a fechadura será rebaixada. Se o intervalo de tempo não tiver expirado, então a bandeira d (demote pendente) será colocada no lugar dela. Isto também programa a máquina de estado para liberar a d (demote pendente) e definir a D (demote) quando o tempo mínimo de espera tiver expirado.

A bandeira I (inicial) é colocada quando a glock tiver sido atribuída a uma fechadura DLM. Isto acontece quando a glock é usada pela primeira vez e a bandeira I permanecerá então definida até que a glock seja finalmente liberada (a qual a trava DLM é desbloqueada).

## 8.5. SUPORTES DE GLOCK

Tabela 8.5, "Bandeiras porta-bandeiras Glock" mostra os significados das diferentes bandeiras porta-glock.

**Tabela 8.5. Bandeiras porta-bandeiras Glock**

| Bandeira | Nome | Significado |
|---|---|---|
| a | Async | Não espere pelo resultado da glock (pesquisará o resultado mais tarde) |
| A | Qualquer | Qualquer modo de bloqueio compatível é aceitável |
| c | Sem cache | Quando desbloqueado, destravar imediatamente a fechadura DLM |
| e | Não expira | Ignorar pedidos subsequentes de cancelamento de bloqueio |
| E | Exato | Deve ter o modo de bloqueio exato |
| F | Primeiro | Definido quando o titular é o primeiro a ser concedido para esta fechadura |
| H | Titular | Indica que o bloqueio solicitado é concedido |
| p | Prioridade | Titular da fila de espera à frente da fila |
| t | Tente | Uma fechadura "try |
| T | Experimente 1CB | Uma fechadura "try" que envia uma chamada de retorno |
| W | Aguarde | Definido enquanto se aguarda a solicitação para completar |

As bandeiras mais importantes são H (suporte) e W (espera), como mencionado anteriormente, uma vez que elas são colocadas em pedidos de fechaduras concedidas e em fila de espera, respectivamente. A ordenação dos detentores na lista é importante. Se houver algum detentor concedido, ele estará sempre à frente da fila, seguido por qualquer detentor em fila de espera.

Se não houver titulares concedidos, então o primeiro titular na lista será aquele que desencadeará a próxima mudança de estado. Como os pedidos de demote são sempre considerados mais prioritários que os pedidos do sistema de arquivos, isso pode nem sempre resultar diretamente em uma mudança no estado solicitado.

O subsistema glock suporta dois tipos de fechadura "try". Estes são úteis tanto porque permitem a retirada de bloqueios fora da ordem normal (com recuo e tentativa adequados) e porque podem ser usados para ajudar a evitar recursos em uso por outros nós. A fechadura t (try) normal é exatamente o que seu nome indica; é uma fechadura "try" que não faz nada de especial. A fechadura T (**try 1CB**), por outro lado, é idêntica à fechadura t, exceto que a DLM enviará uma única chamada de retorno para os atuais porta-fechaduras incompatíveis. Um uso da fechadura T (**try 1CB**) é com as fechaduras **iopen**, que são usadas para arbitrar entre os nós quando a contagem do inode **i_nlink** é zero, e determinar qual dos nós será responsável pela desalocação do inode. A glock **iopen** é normalmente mantida no estado compartilhado, mas quando a contagem de **i_nlink** se torna zero e →**evict_inode**() é chamada, ela solicitará uma fechadura exclusiva com o conjunto T (**try 1CB**). Ela continuará a desalocar o inode se a fechadura for concedida. Se a eclusa não for concedida, resultará no(s) nó(s) que estava(m) impedindo a concessão da eclusa marcando seu(s) glock(s) com a bandeira D (demote), que é verificada em →**drop_inode**() tempo, a fim de garantir que a desalocação não seja esquecida.

Isto significa que os nós que têm contagem zero de ligações, mas ainda estão abertos, serão desalocados pelo nó em que o final **close**() ocorre. Além disso, ao mesmo tempo em que a contagem de links do inodo é decretada a zero, o inodo é marcado como estando no estado especial de ter contagem zero de links, mas ainda em uso no bitmap do grupo de recursos. Isto funciona como a lista de órfãos do sistema de arquivos ext3, pois permite que qualquer leitor subseqüente do bitmap saiba que existe um espaço potencial que pode ser recuperado, e tente recuperá-lo.

## 8.6. RASTROS DE GLOCK

Os tracepoints também são projetados para poder confirmar a exatidão do controle do cache, combinando-os com a saída do **blktrace** e com o conhecimento do layout do on-disk. É então possível verificar se uma determinada E/S foi emitida e completada sob a trava correta, e se não há corridas presentes.

O traço **gfs2_glock_state_change** é o mais importante para entender. Ele rastreia cada mudança de estado da glóculo desde a criação inicial até a despromoção final que termina com **gfs2_glock_put** e o NL final para a transição desbloqueada. A bandeira l (bloqueada) da glock é sempre colocada antes que ocorra uma mudança de estado e só será liberada depois de terminada. Nunca há nenhum detentor concedido (a bandeira H glock holder) durante uma mudança de estado. Se houver algum detentor em fila de espera, ele estará sempre no estado W (em espera). Quando a mudança de estado estiver completa, os titulares poderão ser concedidos, o que é a operação final antes que a bandeira de glock l seja liberada.

O ponto de rastreamento **gfs2_demote_rq** mantém o controle dos pedidos de demote, tanto locais quanto remotos. Assumindo que haja memória suficiente no nó, os pedidos de demote local raramente serão vistos, e na maioria das vezes eles serão criados por **umount** ou ocasionalmente por recuperação de memória. O número de solicitações de demote remoto é uma medida da contenda entre nós para um determinado inode ou grupo de recursos.

O ponto de rastreamento **gfs2_glock_lock_time** fornece informações sobre o tempo gasto pelos pedidos ao DLM. A bandeira de bloqueio (**b**) foi introduzida na glock especificamente para ser usada em combinação com este tracepoint.

Quando um detentor recebe uma fechadura, **gfs2_promote** é chamado, isso ocorre como o estágio final de uma mudança de estado ou quando é solicitada uma fechadura que pode ser concedida imediatamente devido ao estado de glock já cravando uma fechadura de um modo adequado. Se o

suporte for o primeiro a ser concedido para esta glock, então a bandeira f (primeira) é colocada nesse suporte. Isto é usado atualmente apenas por grupos de recursos.

## 8.7. BMAP TRACEPOINTS

O mapeamento de blocos é uma tarefa central para qualquer sistema de arquivos. O GFS2 utiliza um sistema tradicional baseado em bitmap com dois bits por bloco. O objetivo principal dos pontos de rastreamento neste subsistema é permitir o monitoramento do tempo necessário para alocar e mapear blocos.

O ponto de rastreamento **gfs2_bmap** é chamado duas vezes para cada operação de bmap: uma no início para exibir o pedido de bmap, e outra no final para exibir o resultado. Isto facilita a correspondência das solicitações e resultados juntos e mede o tempo necessário para mapear blocos em diferentes partes do sistema de arquivos, diferentes compensações de arquivos, ou mesmo de arquivos diferentes. Também é possível ver quais são os tamanhos médios de extensão que estão sendo devolvidos em comparação aos que estão sendo solicitados.

O **gfs2_rs** tracepoint traça reservas de blocos à medida que são criados, utilizados e destruídos no alocador de blocos.

Para manter o controle dos blocos alocados, **gfs2_block_alloc** é chamado não apenas nas alocações, mas também na liberação de blocos. Como as alocações são todas referenciadas de acordo com o inode ao qual o bloco é destinado, isto pode ser usado para rastrear quais blocos físicos pertencem a quais arquivos em um sistema de arquivos ao vivo. Isto é particularmente útil quando combinado com **blktrace**, que mostrará padrões de E/S problemáticos que podem então ser referidos de volta aos inodes relevantes usando o mapeamento obtido por meio deste tracepoint.

Direct I/O (**iomap**) é uma política de cache alternativa que permite que as transferências de dados de arquivos aconteçam diretamente entre o disco e o buffer do usuário. Isto tem benefícios em situações em que se espera que a taxa de acerto do cache seja baixa. Tanto **gfs2_iomap_start** como **gfs2_iomap_end** tracepoints rastreiam essas operações e podem ser usados para acompanhar o mapeamento usando Direct I/O, as posições no sistema de arquivo do Direct I/O junto com o tipo de operação.

## 8.8. PONTOS DE RASTREAMENTO DE LOG

Os pontos de rastro neste subsistema são adicionados e removidos do diário (**gfs2_pin**), bem como o tempo necessário para comprometer as transações com o diário (**gfs2_log_flush**). Isto pode ser muito útil ao tentar depurar questões de desempenho do periódico.

O ponto de rastreamento **gfs2_log_blocks** mantém registro dos blocos reservados no tronco, o que pode ajudar a mostrar se o tronco é muito pequeno para a carga de trabalho, por exemplo.

O traço **gfs2_ail_flush** é semelhante ao traço **gfs2_log_flush**, na medida em que mantém o controle do início e do fim dos fluxos da lista AIL. A lista AIL contém buffers que já passaram pelo log, mas ainda não foram escritos de volta no lugar e isto é periodicamente lavado a fim de liberar mais espaço de log para uso pelo sistema de arquivo, ou quando um processo solicita um **sync** ou **fsync**.

## 8.9. ESTATÍSTICAS DA GLOCK

O GFS2 mantém estatísticas que podem ajudar a rastrear o que está acontecendo dentro do sistema de arquivos. Isto permite detectar problemas de desempenho.

O GFS2 mantém dois balcões:

- **dcount**, que conta o número de operações DLM solicitadas. Isto mostra quantos dados entraram nos cálculos da média/variância.
- **qcount**, que conta com o número de operações de nível **syscall** solicitadas. Geralmente **qcount** será igual ou maior que **dcount**.

Além disso, o GFS2 mantém três pares de média/variância. Os pares de média/variância são estimativas exponenciais suavizadas e o algoritmo utilizado é o utilizado para calcular os tempos de ida e volta em código de rede.

Os pares de média e variância mantidos no GFS2 não são escalados, mas estão em unidades de nanossegundos inteiros.

- srtt/srttvar: Tempo de ida e volta suavizado para operações sem bloqueios
- srttb/srttvarb: Tempo suavizado de ida e volta para operações de bloqueio
- irtt/irttvar: Tempo entre pedidos (por exemplo, tempo entre pedidos de DLM)

Um pedido sem bloqueio é aquele que será concluído imediatamente, qualquer que seja o estado da fechadura DLM em questão. Isso significa atualmente qualquer solicitação quando (a) o estado atual da fechadura é exclusivo (b) o estado solicitado é nulo ou desbloqueado ou (c) a bandeira "try lock" é colocada. Um pedido de bloqueio cobre todos os outros pedidos de bloqueio.

Tempos maiores são melhores para os IRTTs, enquanto que tempos menores são melhores para os RTTs.

As estatísticas são mantidas em dois arquivos **sysfs**:

- O arquivo **glstats**. Este arquivo é similar ao arquivo **glocks**, exceto que contém estatísticas, com uma glock por linha. Os dados são inicializados a partir dos dados "por cpu" para aquele tipo de glock para o qual a glock é criada (além dos contadores, que são zerados). Este arquivo pode ser muito grande.
- O arquivo **lkstats**. Este contém as estatísticas "por cpu" para cada tipo de glock. Ele contém uma estatística por linha, na qual cada coluna é um núcleo cpu. Há oito linhas por tipo de glóculo, com tipos que se sucedem entre si.

## 8.10. REFERÊNCIAS

Para mais informações sobre tracepoints e o arquivo GFS2 **glocks**, consulte os seguintes recursos:

- Para informações sobre as regras de bloqueio interno glock, veja https://git.kernel.org/pub/scm/linux/kernel/git/torvalds/linux.git/tree/Documentation/filesystem glocks.rst.
- Para informações sobre o rastreamento de eventos, veja https://git.kernel.org/pub/scm/linux/kernel/git/torvalds/linux.git/tree/Documentation/trace/eve
- Para informações sobre a utilidade **trace-cmd**, veja http://lwn.net/Articles/341902/.

# CAPÍTULO 9. MONITORAMENTO E ANÁLISE DOS SISTEMAS DE ARQUIVO GFS2 USANDO O PERFORMANCE CO-PILOT (PCP)

Esta seção fornece informações sobre o uso do Performance Co-Pilot (PCP) para ajudar no monitoramento e na análise dos sistemas de arquivos GFS2. O monitoramento dos sistemas de arquivo GFS2 no PCP é fornecido pelo módulo GFS2 PMDA no Red Hat Enterprise Linux, que está disponível através do pacote **pcp-pmda-gfs2**.

O GFS2 PMDA fornece uma série de métricas fornecidas pelas estatísticas do GFS2 fornecidas no subsistema **debugfs**. Quando instalado, o PMDA expõe valores fornecidos nos arquivos **glocks**, **glstats**, e **sbstats**. Estes conjuntos de relatórios de estatísticas sobre cada sistema de arquivos GFS2 montado. O PMDA também faz uso dos pontos de rastreamento do kernel GFS2 expostos pelo Kernel Function Tracer (**ftrace**).

## 9.1. INSTALANDO O GFS2 PMDA

Para operar corretamente, o GFS2 PMDA requer que o sistema de arquivo **debugfs** esteja montado. Se o sistema de arquivo **debugfs** não estiver montado, execute os seguintes comandos antes de instalar o GFS2 PMDA:

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

O GFS2 PMDA não está habilitado como parte da instalação padrão. Para fazer uso do monitoramento métrico GFS2 através do PCP, você deve habilitá-lo após a instalação.

Execute os seguintes comandos para instalar o PCP e habilitar o GFS2 PMDA. Note que o script de instalação do PMDA deve ser executado como root.

```
# yum 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
```

## 9.2. EXIBINDO INFORMAÇÕES SOBRE AS MÉTRICAS DE DESEMPENHO DISPONÍVEIS COM A FERRAMENTA PMINFO

A ferramenta **pminfo** exibe informações sobre as métricas de desempenho disponíveis. As seções seguintes mostram exemplos de diferentes métricas GFS2 que você pode exibir com esta ferramenta.

### 9.2.1. Examinando o número de estruturas de glock que existem atualmente por sistema de arquivo

As métricas de glock GFS2 dão uma visão do número de estruturas de glock atualmente incore para cada sistema de arquivo GFS2 montado e seus estados de travamento. No GFS2, uma glock é uma estrutura de dados que reúne o DLM e o caching em uma única máquina de estado. Cada glock tem um

mapeamento 1:1 com uma única trava DLM e fornece cache para os estados de travamento de modo que operações repetitivas realizadas em um único nó não tenham que chamar repetidamente o DLM, reduzindo o tráfego desnecessário na rede.

O seguinte comando **pminfo** exibe uma lista do número de glocks por sistema de arquivo GFS2 montado através de seu modo de bloqueio.

```
# 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
```

### 9.2.2. Examinando o número de estruturas de glock que existem por sistema de arquivo, por tipo

As métricas de glstats GFS2 dão contagens de cada tipo de glock que existem para cada arquivo ystem, um grande número destes normalmente será do tipo inode (inode e metadados) ou de grupo de recursos (metadados de grupo de recursos).

O seguinte comando **pminfo** exibe uma lista do número de cada tipo de Glock por sistema de arquivo GFS2 montado.

```
# 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
```

### 9.2.3. Verificação do número de estruturas de glock que estão em estado de espera

As bandeiras de suporte mais importantes são H (suporte: indica que o bloqueio solicitado é concedido) e W (aguarde: definido enquanto se aguarda o pedido para concluir). Estas bandeiras são colocadas em pedidos de fechaduras concedidas e pedidos de fechaduras em fila de espera, respectivamente.

O seguinte comando **pminfo** exibe uma lista do número de glocks com a bandeira de suporte Wait (W) para cada sistema de arquivo GFS2 montado.

```
# 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
```

Se você vir uma série de pedidos em fila de espera em um bloqueio de grupo de recursos, pode haver uma série de razões para isso. Uma delas é que há um grande número de nós em comparação com o número de grupos de recursos no sistema de arquivos. Outra é que o sistema de arquivos pode estar muito próximo de estar cheio (exigindo, em média, buscas mais longas por blocos livres). A situação em ambos os casos pode ser melhorada adicionando mais armazenamento e usando o comando **gfs2_grow** para expandir o sistema de arquivo.

### 9.2.4. Verificação da latência de operação do sistema de arquivo usando a métrica baseada no ponto de traço do kernel

O GFS2 PMDA suporta a coleta de métricas a partir dos pontos de rastro do núcleo do GFS2. Por padrão, a leitura dessas métricas é desativada. A ativação dessas métricas ativa os pontos de rastro do kernel GFS2 quando as métricas são coletadas a fim de preencher os valores das métricas. Isto poderia ter um pequeno efeito no rendimento de desempenho quando estas métricas de pontos de traço do kernel são ativadas.

O PCP fornece a ferramenta **pmstore**, que permite modificar as configurações do PMDA com base em valores métricos. A métrica **gfs2.control.*** permite a troca de pontos de rastreamento de kernel GFS2.

O exemplo a seguir usa o comando **pmstore** para habilitar todos os pontos de rastreamento do kernel GFS2.

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

Quando este comando é executado, o PMDA liga todos os tracepoints GFS2 no sistema de arquivo **debugfs**. Tabela 9.1, “Lista métrica completa” explica cada um dos tracepoints de controle e sua utilização, Uma explicação sobre o efeito de cada tracepoint de controle e suas opções disponíveis também está disponível através da chave de ajuda em **pminfo**.

O GFS2 promove métricas que contam o número de solicitações de promoção no sistema de arquivos. Estas solicitações são separadas pelo número de solicitações que ocorreram na primeira tentativa e "outras" que são concedidas após sua solicitação inicial de promoção. Uma queda no número de promoções da primeira vez com um aumento em "outras" promoções pode indicar problemas com a contenção de arquivos.

As métricas de solicitação de demota GFS2, assim como as métricas de solicitação de promoção, contam o número de solicitações de demota que ocorrem no sistema de arquivo. Estes, entretanto, também são divididos entre as solicitações que vieram do nó atual e as solicitações que vieram de outros nós do sistema. Um grande número de solicitações demoteadas de nós remotos pode indicar a contenção entre dois nós para um determinado grupo de recursos.

A ferramenta **pminfo** exibe informações sobre as métricas de desempenho disponíveis. Este procedimento exibe uma lista do número de glocks com a bandeira de suporte Wait (W) para cada sistema de arquivo GFS2 montado. O seguinte comando **pminfo** exibe uma lista do número de glocks com a bandeira de suporte Wait (W) para cada sistema de arquivo GFS2 montado.

```
# 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
```

É uma boa idéia determinar os valores gerais observados quando a carga de trabalho está funcionando sem problemas para poder perceber mudanças no desempenho quando esses valores diferem de sua faixa normal.

Por exemplo, você pode notar uma mudança no número de pedidos de promoção esperando para completar em vez de completar na primeira tentativa, o que o resultado do comando seguinte lhe permitiria determinar.

```
# 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
```

A saída do comando seguinte permitiria determinar um grande aumento nas solicitações de demote remoto (especialmente se de outros nós de 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
```

A saída do comando seguinte poderia indicar um aumento inexplicável dos fluxos de toras.

```
# 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
```

## 9.3. LISTA COMPLETA DE MÉTRICAS DISPONÍVEIS PARA GFS2 EM PCP

Tabela 9.1, "Lista métrica completa" descreve a lista completa das métricas de desempenho fornecidas pelo pacote **pcp-pmda-gfs2** para os sistemas de arquivos GFS2.

**Tabela 9.1. Lista métrica completa**

| Nome métrico | Descrição |
|---|---|
| **gfs2.glocks.*** | Métricas relativas às informações coletadas do arquivo de estatísticas de glock (**glocks**) que contam o número de glocks em cada estado que existe atualmente para cada sistema de arquivo GFS2 atualmente montado no sistema. |

| Nome métrico | Descrição |
| --- | --- |
| **gfs2.glocks.flags.*** | Gama de métricas contando o número de glocks que existem com as bandeiras de glocks dadas |
| **gfs2.holders.*** | Métricas relativas às informações coletadas do arquivo de estatísticas de glock (**glocks**) que conta o número de glocks com suportes em cada estado de bloqueio que existe atualmente para cada sistema de arquivo GFS2 atualmente montado no sistema. |
| **gfs2.holders.flags.*** | Gama de métricas contando o número de porta-bandeiras com as bandeiras de suporte dadas |
| **gfs2.sbstats.*** | Métricas de tempo relativas às informações coletadas do arquivo de estatísticas do superbloco (**sbstats**) para cada sistema de arquivos GFS2 atualmente montado no sistema. |
| **gfs2.glstats.*** | Métricas referentes às informações coletadas do arquivo de estatísticas da glock (**glstats**) que contam o número de cada tipo de glock que existe atualmente para cada sistema de arquivo GFS2 atualmente montado no sistema. |
| **gfs2.latency.grant.*** | Uma métrica derivada fazendo uso dos dados do **gfs2_glock_queue** e **gfs2_glock_state_change** tracepoints para calcular uma latência média em microssegundos para pedidos de concessão de subsídios de glock a serem completados para cada sistema de arquivo montado. Esta métrica é útil para descobrir possíveis lentidão no sistema de arquivo quando a latência da concessão aumenta. |
| **gfs2.latency.demote.*** | Uma métrica derivada fazendo uso dos dados do **gfs2_glock_state_change** e **gfs2_demote_rq** tracepoints para calcular uma latência média em microssegundos para pedidos de demote de glock a serem completados para cada sistema de arquivo montado. Esta métrica é útil para descobrir possíveis lentidão no sistema de arquivo quando a latência demotada aumenta. |
| **gfs2.latency.queue.*** | Uma métrica derivada fazendo uso dos dados do ponto de rastreamento **gfs2_glock_queue** para calcular uma latência média em microssegundos para pedidos de fila de glock a serem completados para cada sistema de arquivo montado. |

| Nome métrico | Descrição |
|---|---|
| **gfs2.worst_glock.*** | Uma métrica derivada fazendo uso dos dados do traço **gfs2_glock_lock_time** para calcular uma percepção de "pior glock atual" para cada sistema de arquivo montado. Esta métrica é útil para descobrir uma possível contenção da fechadura e uma desaceleração do sistema de arquivo se a mesma fechadura for sugerida várias vezes. |
| **gfs2.tracepoints.*** | Métricas relativas à saída do GFS2 **debugfs** tracepoints para cada sistema de arquivo atualmente montado no sistema. Cada subtipo dessas métricas (um de cada ponto de rastreamento GFS2) pode ser controlado individualmente, seja ligado ou desligado, utilizando as métricas de controle. |
| **gfs2.control.*** | Métricas de configuração que são usadas para ligar ou desligar o registro métrico no PMDA. As métricas de patrulha são alternadas por meio da ferramenta **pmstore**. |

## 9.4. REALIZAR A CONFIGURAÇÃO MÍNIMA DE PCP PARA COLETAR DADOS DO SISTEMA DE ARQUIVOS

O procedimento a seguir descreve instruções sobre como instalar uma configuração mínima de PCP para coletar estatísticas sobre o Red Hat Enterprise Linux. Esta configuração envolve a adição do número mínimo de pacotes em um sistema de produção necessários para coletar dados para análise posterior.

O arquivo **tar.gz** resultante da produção do **pmlogger** pode ser analisado usando ferramentas adicionais de PCP e pode ser comparado com outras fontes de informação de desempenho.

1. Instalar os pacotes de PCP necessários.

   ```
   # yum install pcp pcp-pmda-gfs2
   ```

2. Ativar o módulo GFS2 para PCP.
   **# cd /var/lib/pcp/pmdas/gfs2 ./Install**

3. Iniciar os serviços **pmcd** e **pmlogger**.
   **# systemctl start pmcd.service systemctl start pmlogger.service**

4. Realizar operações no sistema de arquivos GFS2.

5. Pare os serviços **pmcd** e **pmlogger**.
   **# systemctl stop pmcd.service systemctl stop pmlogger.service**

6. Colete a saída e salve-a em um arquivo **tar.gz** nomeado com base no nome do host e na data e hora atuais.

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

## 9.5. REFERÊNCIAS

- Para mais informações sobre o monitoramento de desempenho do GFS2 em geral, veja Depuração de sistemas de arquivo GFS2 com pontos de rastreamento GFS2 e o arquivo de depuração de glocks GFS2.
- Para informações mais gerais sobre o uso do Performance Co-Pilot para monitorar o desempenho do sistema, veja Monitorando o desempenho com o Performance Co-Pilot no Portal do Cliente da Red Hat.
- Para informações gerais sobre o Co-Piloto de Desempenho no Red Hat Enterprise Linux veja os artigos, soluções, tutoriais e white papers do Index of Performance Co-Pilot(PCP) no Portal do Cliente da Red Hat.