

## Workshop: Ferramentas para manipulação de sistemas de arquivos

# Oficina de formatação

A manutenção de sistemas de arquivos não é mais uma ação necessariamente demorada. Tanto o JFS, quanto o ReiserFS e o XFS lidam bem com grandes partições. O JFS e o XFS não precisam de mais que um piscar de olhos para formatar um disco rígido de 1 TByte, operação que o ReiserFS leva meio minuto para fazer. Apenas o Ext 3 ainda deixa o usuário esperando por cerca de uma hora, o que não deveria ser de admirar, afinal, no fundo ele continua sendo um sistema de arquivos Ext 2, equipado com recursos de journaling.

Além dos programas de formatação, existem ainda muitos outros utilitários para cada sistema de arquivos. Alguns deles podem ajudar a melhorar a performance. Outros são usados somente na ocorrência de erros. A seguir apresentamos os comandos mais importantes. Elas são utilizadas ou a partir da linha de comando com o comando *mount* ou de alterações efetuadas pelo usuário no arquivo */etc/fstab*.

### Ext 2/3: rico em opções

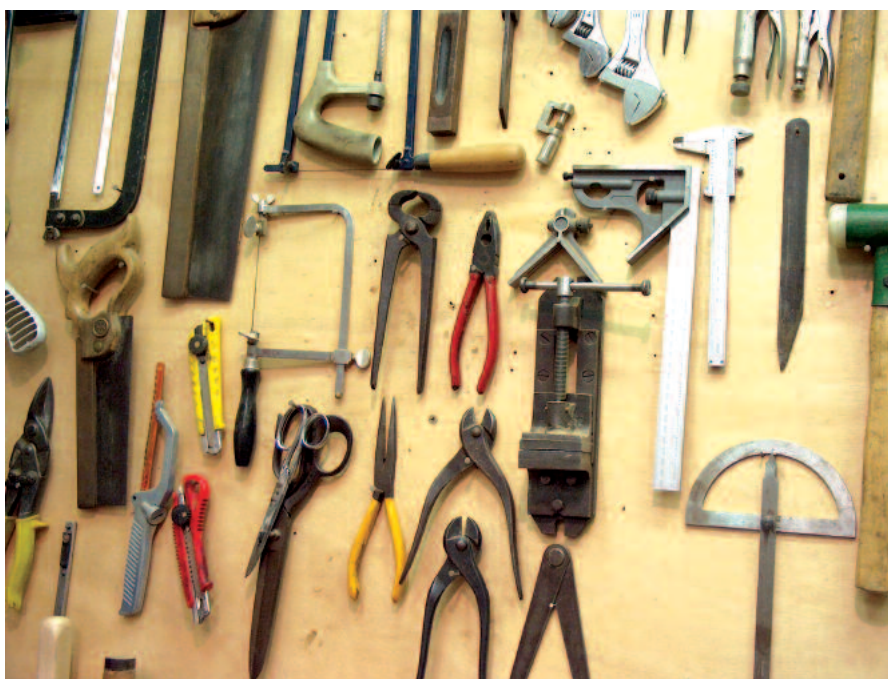
É uma boa idéia equipar um sistema de arquivos Ext 2 com um journal, e atualizá-lo assim para Ext 3. Todo o processo é simples e não causa problemas; pelo menos os usuários nunca relataram nada nos fóruns específicos. Ainda assim, nunca é uma má idéia fazer um backup de todos os seus dados importantes antes da operação.

Para usar o Ext 3, o suporte a Ext 3 precisa estar ativado no kernel. Se você equipa a partição root com Ext 3, o driver do Ext3 deve ser, via de regra, compilado diretamente no kernel. Se se trabalha com um disco de RAM (initrd), entretanto, pode-se compilar o driver como módulo. A partir do momento em que o suporte a Ext 3 está disponível, a atualização pode começar. Para esta finalidade basta utilizar o programa *tune2fs*, contido no pacote *e2fsprogs* [1]:

```
tune2fs -j /dev/hda2
```

Os programas de formatação atuais fazem muito mais do que apenas criar sistemas de arquivos. Como caixas de ferramentas bem organizadas, eles oferecem um utensílio ideal para cada tarefa. Quem quiser aumentar o desempenho do seu sistema pode arrancar até o último bit de desempenho do Ext 3, JFS, ReiserFS ou XFS se utilizar corretamente as opções do comando *mount*.

POR JÖRG REITTER



A opção *-j* cria o journal e o coloca no arquivo *.journal* no diretório raiz da partição. Para concluir, modifique o tipo do sistema de arquivos de Ext 2 para Ext 3 no arquivo */etc/fstab*. Esta é uma das raras ocasiões em que excepcionalmente é mais simples reinicializar o sistema para que as modificações surtam efeito. Se depois você quiser converter o sistema de arquivos novamente em Ext 2, basta desmontá-lo e montá-lo novamente com o tipo *ext2*. Se quiser tornar isso permanente, modifique o tipo do sistema de arquivos novamente, alterando o arquivo */etc/fstab*.

Quanto maior o sistema de arquivos, mais tempo leva a verificação de integri-

dade, que acontece a intervalos pré-determinados. Caso os intervalos sejam muito espaçados ou curtos demais (por exemplo, como no caso de testes intensivos), este comportamento pode ser alterado de acordo com as necessidades do usuário. Com a opção *-i* do *e2fsck* ajusta-se os dias, semanas ou meses nos quais o programa deve executar a verificação de integridade do sistema de arquivos, que pode também depender do número de vezes que ele é montado – o parâmetro *-c* é usado para configurar o sistema desta forma. O argumento *0*, usado em ambas as opções, desativa completamente a verificação do sistema de arquivos.

## Pode formatar!

Para começar, o ideal é realizar alguns testes em uma partição pequena. Sem opções adicionais, o comando de formatação `mkfs.ext3` determina os parâmetros necessários automaticamente. Assim, é relativamente simples formatar uma partição. Digite:

```
mkfs.ext3 /dev/sdb2
```

O tamanho de bloco apropriado é calculado pelo programa a partir do tamanho da partição. Se este não for o comportamento desejado, o próprio usuário pode fornecer o tamanho do bloco a ser utilizado (1,2 ou 4 kByte):

```
mkfs.ext3 -b 2048 /dev/sdb2
```

O `mkfs.ext3` oferece também opções com finalidades específicas. Com a opção `-T` o usuário ajusta o desempenho do sistema de arquivos de acordo com o tamanho dos arquivos a ser manipulados. Baseado nisso, o programa determina as configurações ideais. Os tamanhos disponíveis são 4 KBytes (*news*), 1 MByte (*largefile*) e 4 MBytes (*largefile4*). O comando a seguir parte do princípio de que a maioria dos arquivos do sistema de arquivos terão 1 MByte:

```
mkfs.ext3 -T largefile
```

## Blocos malvados!

Nunca é demais verificar a integridade do disco rígido antes de instalar o sistema de arquivos. A opção `-c` instrui o programa `mkfs.ext3` a examinar o disco à procura de blocos ruins – os famosos “bad blocks”. Esta opção deve ser aplicada principalmente em hardware mais antigo. Para saber exatamente como o meio físico do disco está, pode-se especificar a opção `-c` duas vezes seguidas, o que instrui o `mkfs.ext3` a executar um teste (**destrutivo!**) de leitura e escrita (“read-write”), ao invés de somente um teste de leitura (“read”). Entretanto, este tipo de teste, além de arriscado, demora duas vezes mais para terminar.

O `mkfs.ext3` também conta com opções de journal. O usuário seleciona se o journal deve ser armazenado internamente no superbloco (`size = 1024`) ou externamente, em outro dispositivo (como em `device = /dev/hdc2`). No caso

## Ext 3: opções do comando `mount`

Você encontra informações detalhadas sobre o sistema de arquivos Ext 3 em

```
/usr/src/linux/Documentation/filesystems/ext3.txt.
```

*Nota:* O Ext 2 ignora opções de quota como `grpquota`, `noquota`, `quota` e `usrquota` sem mostrar mensagens de erro.

**data=journal:** Os metadados vão primeiro para o journal, e somente então são escritos no sistema de arquivos (metadados e journaling de dados).

**data=ordered:** Primeiro os dados são escritos no sistema de arquivos; somente então os metadados são adaptados.

**data=writeback:** Os dados “reais” podem ser escritos no sistema de arquivos antes ou depois da atualização do journal de metadados.

**journal=update:** Atualiza o journal para o formato corrente.

**journal=inum:** Com `inum` o super-usuário fornece o número do inode que representa o arquivo do journal (ignorado, caso já exista um journal).

**noload:** O Journal não é carregado.

**bsddf:** O comando `df` reage como no BSD.

**minixdf:** O comando `df` reage como no Minix  
**check=none, nocheck:** Não faz nenhuma verificação suplementar de bitmaps no decorrer da montagem.

**debug:** O kernel envia informações suplementares de depuração ao Syslog.

**errors=continue:** A montagem é levada adiante, mesmo que ocorra um erro no sistema de arquivos.

**errors=remount-ro:** No caso de erro, o sistema de arquivos é montado novamente apenas em modo de leitura.

**errors=panic:** A máquina pára no caso de erro no sistema de arquivos.

**grpuid, bsdgroups:** Dá a um objeto a mesma ID de grupo do objeto pai.

**nogrpuid, sysvgroups:** Objetos novos recebem a ID de grupo do criador.

**resuid=UID:** Aqui ficam as IDs de usuários que têm acesso a blocos reservados

**resgid=GID:** Aqui ficam as IDs de grupos que podem utilizar blocos reservados.

**sb=offset:** Utiliza um superbloco alternativo, encontrado na posição indicada em `offset`.

da variante externa, o journal já deve existir antes que o usuário crie o sistema de arquivos via `mkfs.ext3`. Neste caso, dois passos são necessários: em primeiro lugar devemos formatar a partição onde o journal ficará. Ela deve ter o mesmo tamanho de bloco (1, 2 ou 4 KBytes) do sistema de arquivos que vai utilizar o journal nela armazenado. O comando a seguir gera o journal para um sistema de arquivos em `/dev/hdb1` no dispositivo externo `/dev/hda9`:

```
mkfs.ext3 -O /dev/hdb1 /dev/hda9
```

Para informar ao sistema de arquivos em `/dev/hdb1` onde encontrar o journal externo, usa-se o parâmetro `-J` do comando `tune2fs`:

```
tune2fs -J device=/dev/hda9
```

Após o usuário determinar todas as opções para criar o sistema de arquivos de acordo com as suas necessidades, a opção `-n` pode ser utilizada para simular a criação do sistema de arquivos. Deste modo, o usuário vê imediatamente se as opções escolhidas são adequadas.

Existem ainda outras opções que podem ser interessantes em casos especiais. Por exemplo, uma delas reserva mais de 5% do espaço disponível na partição para o usuário root (`-m`), enquanto outra desativa funções que são ativadas por padrão (`-O`).

Outros programas importantes são o `dumpe2fs`, que mostra informações sobre o sistema de arquivos, fornecendo o conteúdo do superbloco e dos grupos de bloco, ou o `e2fsck`, que verifica a integridade de sistemas de arquivos Ext 2 e Ext 3, à procura de erros. O `e2fsck` normalmente não precisa ser utilizado com um sistema Ext 3, a única exceção é quando o superbloco está danificado e necessita de verificação. O uso do LVM (veja o artigo na página 38 desta edição) permite que se aumente ou diminua, sem perda de dados, o tamanho de uma partição com o utilitário `e2fsadm`, que controla o redimensionamento on-line e off-line. Os recursos de “dump” e “restore” do Ext 2 continuam funcionando também com o Ext 3, de modo que o usuário ou o administrador do sistema não precisa mudar seus hábitos, nem suas ferramentas.

## ReiserFS: Opções de montagem

Você encontra informações detalhadas sobre o sistema de arquivos ReiserFS em

<http://www.namesys.com/mount-options.html>

**conv:** Com ele o ReiserFS versão 3.6 pode montar um sistema de arquivos versão 3.5, utilizando o código da versão 3.6 para os novos objetos criados. Depois disso, as ferramentas da versão 3.5 do ReiserFS não funcionam mais para este sistema de arquivos.

**dontpanic (somente para reiserfs-raw):**

Ignora erros de entrada e saída enquanto o journal está sendo processado. Somente disponível em modo *reiserfs-raw*.

**hash=rupasov:** Algoritmo de hash muito rápido para diretórios enormes e nomes de arquivos incomuns. Perigosa de usar, uma vez que colisões hash são bastante prováveis. Colisão hash é o nome dado à situação que ocorre quando dois argumentos diferentes em uma função de hash produzem resultados idênticos.

**hash=tea:** Altamente aleatório e com baixa probabilidade de colisões hash, mas reduz o desempenho. Deve ser utilizada caso a mensagem EHASHCOLLISION ocorra e o algoritmo de hash *r5* esteja sendo usado.

**hash=r5:** Hash padrão.

**hash=detect:** A função hash é reconhecida automaticamente quando o sistema de arquivos é montado e escrita no superbloco.

**hashed\_relocation:** Procura ajustar o alocador de bloco. Pode melhorar a performance. Ver também *no\_unhashed\_relocation*.

**no\_unhashed\_relocation:** Procura ajustar o alocador de bloco. Pode melhorar a performance. Ver também *hashed\_relocation*.

**noborder:** Desativa o *border allocator algorithm*. Pode aumentar o desempenho.

**nolog:** Desativa o Journaling. Pode melhorar a performance. Cuidado: isto desativa a recuperação rápida após um crash.

**notail:** Desativa a ferramenta *tail*. Se ocorrerem problemas com um programa durante sua operação normal (como o *lilo*, por exemplo), o ReiserFS não deverá armazenar arquivos minúsculos diretamente na árvore.

**pgc=LOW, HIGH:** Ativa o "coletor de lixo passivo" (Passive Garbage Collector). Só funciona em conjunto com o driver *reiserfs-raw*, que não está no kernel padrão.

**raw:** Monta o sistema de arquivos no modo *raw*. Não disponível no kernel padrão.

**replayonly:** Reverte as transações no journal sem montar o sistema de arquivos. Geralmente, somente o *fsck* utiliza esta opção.

**resize=valor:** Aumenta o tamanho de uma partição que está montada, em particular dispositivos LVM (Logical Volume Management). Requer a Resizer-Utility, que pode ser baixada do servidor de FTP da Namesys: [ftp.namesys.com/pub/reiserfsprogs](http://ftp.namesys.com/pub/reiserfsprogs).

```
mkfs.jfs -J device=external z
-journal device
```

O dispositivo (*device*) no final da linha de comando acima é a partição na qual o sistema de arquivos será criado (ou seja, algo como */dev/hd?n*). Num mesmo comando se pode usar ou *journal\_dev* ou *external-journal*; não ambos ao mesmo tempo. Quem precisa de compatibilidade com o OS/2 deve usar a opção *-O*, para que o sistema se torne insensível a maiúsculas e minúsculas. A dimensão dos blocos é fixa em 4096 bytes. Quanto a isto o usuário nada pode fazer.

Há apenas mais três outras aplicações disponíveis para o JFS: com *jfs\_tune* pode-se, entre outras coisas, listar o nome do volume e sua UUID. O comando *jfs\_debugfs* executa diversas ações de baixo nível, como substituir dados em um offset com uma string hexadecimal, para fins de depuração de problemas. Por fim, *jfs\_fscklog* extrai e mostra o conteúdo dos relatórios do *fsck*.

## Reiser pé-de-chumbo

O ReiserFS v3.6 é, ao mesmo tempo, um sistema de arquivos estável e um ótimo brinquedo para os famintos por performance. O conjunto de programas *Reiser fsprogs* [3] contém vários utilitários com opções de ajuste e ferramentas de recuperação de erros. Por outro lado, não existem programas de dump/restore. É possível, entretanto, aumentar e diminuir o tamanho do sistema de arquivos.

Ainda que o ReiserFS seja o sistema de arquivos padrão do SuSE Linux, não há qualquer informação sobre ele disponível na distribuição. A única coisa que existe é o programa de formatação *mkfs.reiserfs*. Não há sequer uma página de manual ou qualquer sinal de documentação no diretório */usr/share/doc*. O interessante é que nem mesmo na documentação do kernel aparece alguma referência. Só resta ir à Internet e se dirigir à homepage do líder do projeto, Hans Reiser [3]. A visita ao site vale a pena: lá se encontram patches especiais para os corajosos de plantão, e os membros da lista de discussão respondem a muitas questões, ajudando a eliminar dúvidas.

Um novo sistema de arquivos é criado com a instrução *mkreiserfs*. Até existe uma opção de ajuste de tamanho de bloco, mas somente a partir da recém-

## JFS: Sovina em opções

A versão do JFS para Linux atingiu um estágio que já permite a realização de testes sem dificuldades. A inadequação atual para o uso em ambientes de produção (pelo menos rodando em Linux) fica clara pela curta lista de opções. Por outro lado, os resultados do testes de desempenho aplicados ao JFS atingiram valores excelentes (ver artigo a partir da página 23). Para análises detalhadas há uma interface de depuração no código do kernel. Entretanto, no que tange a programas que possam interagir com o JFS a nível de usuário (i.e., em "user space"), o administrador de sistemas só dispõe da ferramenta *jfs\_fscklog* do pacote *Jfsutils* [2], que extrai o conteúdo do arquivo de registro gerado pelo *fsck* e o exibe no console.

Comparativamente, o JFS é pobre em de opções a nível de usuário. Assim, não é possível, por exemplo, ajustar o tamanho de bloco. Entretanto, as pági-

nas de manual (*man pages*) são bem cuidadas e fornecem bons exemplos, ao contrário da documentação disponível para os outros sistemas de arquivos. Algumas opções do *mount* visam igualmente o tratamento de erros, e fornecem informações importantes para as listas de discussões.

## Cadê as aplicações?

O comando *mkfs.jfs* cria um sistema de arquivos, e com o parâmetro *-c* o usuário pode procurar por blocos. Com o JFS também é possível colocar o journal em outro dispositivo (*-J*). O comando:

```
mkfs.jfs -J journal_dev z
external-journal
```

cria o journal no dispositivo indicado. Subseqüentemente, a ligação entre o journal e o sistema de arquivos, que agora será realmente gerado, ocorria via *device*, através do comando:

lançada versão 4 o usuário pode selecionar seu tamanho. O ReiserFS seleciona outras opções sozinho, caso elas não sejam especificadas, tais como algoritmos de hash (*r5*, *rupasov*, *tea*) ou se a versão 3.5 ou a 3.6 deve ser instalada.

As opções para o journal se restringem ao tamanho do journal (-s), se ele se encontra num dispositivo externo (-j) e a partir de qual offset ele começa (-o). Além disso, o usuário pode ajustar a dimensão máxima da transação – da mesma forma que ocorre em uma transação de um banco de dados, transações de sistemas de arquivos com journaling tratam seqüências de modificações dos metadados do sistema de como uma única operação atômica, ou seja, ou ela ocorre inteiramente ou simplesmente não ocorre. O ajuste do tamanho da transação é sempre seguro, uma vez que o *mkreiserfs* o adapta automaticamente, caso o usuário use um valor incorreto. Quem utiliza o UUID (Universally Unique Identifier), pode ajustá-lo com a opção *-u*. Sem esta opção, o *mkreiserfs* escolhe o valor ideal automaticamente.

## ReiserFS na oficina

Logo após a inicialização do sistema, entra em cena o programa de verificação de integridade e reparos do sistema de arquivos, *reiserfsck*. Na ocorrência de um crash, ele controla a consistência do sistema de arquivos, usando a opção padrão (*check*) e restabelece o último estado do sistema armazenado no journal. Se o *check* constata um erro grave, tal como a inconsistência de um sistema de arquivos, ele pede ao usuário para que tome as providências necessárias, e mostra no console a opção correta a ser usada. Para inconsistências pequenas, como nomes de diretórios inválidos ou truncados, ele recomenda usar a opção *fix-fixable*. Entretanto, se o sistema de arquivos foi corrompido de forma mais grave, o usuário deve tentar a opção *rebuild-tree*. O *reiserfsck* irá reconstruir toda a árvore do sistema de arquivos, mas isto leva muito tempo. E, piorando a situação, ainda há risco de, ao final da operação, o sistema de arquivos estar completamente destruído.

Se aparecer a mensagem de erro *read\_super\_block: cant't find a reiserfs file system*, o usuário executar o comando:

## XFS: Opções de montagem

Você encontra informações detalhadas sobre o sistema de arquivos XFS em */usr/src/linux/Documentation/filesystems/xfs.txt*.

**biosize=valor:** Configura o tamanho preferido de I/O com buffer (o padrão é 64 kBytes). O argumento *valor* deve ser expresso como o logaritmo de base 2 do tamanho de I/O desejado.

**ikeep:** Mantém grupos de inodos vazios, em vez de devolvê-los ao pool de memória livre.

**logbufs= valor:** Define o número de buffers de registro (log) na memória central. Pode-se usar valores de 2 a 8. Pode melhorar o desempenho do sistema, mas reduz a quantidade de memória livre.

**logbsize=valor:** Define o tamanho dos buffers de registro (log). Pode-se usar os valores 16, 32, 64, 128 e 256 kBytes. Sistemas com mais de 32 MBytes de RAM usam 32 kBytes por padrão.

**logdev=dispositivo** Cria *journals* externos.

**noalign:** As alocações de dados não são alinhadas às bordas de "stripe units".

**noatime:** Não atualiza os *timestamps* durante acessos de leitura.

**norecovery:** Monta o sistema de arquivos sem a recuperação de registro (Log-Recovery). O sistema de arquivos deve ser montado em modo somente de leitura.

**osyncisync:** Atualiza os timestamps dos arquivos que tem o flag *O\_SYNC* ativado. Sem esta opção a performance deve ser melhor. Entretanto, os arquivos perdem a informação de timestamp no caso de um crash.

**quota, usrquota, uqnoenforce:** Ativa o sistema de quotas para usuários, pode forçar que os limites sejam respeitados.

**grpquota, gqnoenforce:** Ativa o sistema de quotas para grupos e pode forçar que os limites sejam respeitados.

**sunit=valor e swidth= valor:** Estabelece o tamanho da stripe unit em intervalos de 512 bytes (unidades de bloco). A opção *swidth* deve ser configurada um múltiplo de valor de *sunit*.

*Importante para dispositivos RAID:* caso se modifique o layout do disco depois que o sistema de arquivos tenha sido criado, as informações podem ser sobrescritas no superbloco pelo usuário *root*.

**nouuid:** Não verificar via UUID se o sistema de arquivos for duplamente montado. Útil na montagem do LVM-Snapshots.3

## JFS: Opções de montagem

Você encontra informações detalhadas sobre o sistema de arquivos JFS em */usr/src/linux/Documentation/filesystems/jfs.txt*.

**iocharset=nome:** Conjunto de caracteres que o sistema deve utilizar para converter de Unicode para ASCII.

**resize=n:** Aumenta o tamanho do volume online em *n* blocos.

**nointegrity:** Evita que o journal seja escrito. Interessante quando o backup de um volume precisa ser restaurado.

**integrity:** Modificações nos metadados são protocoladas pelo journal. Deve ser utilizada após restaurar um backup com a opção *nointegrity*, em conjunto com uma remontagem.

**errors=continue:** A montagem continua normalmente, mesmo se ocorrer um erro no sistema de arquivos.

**errors=remount-ro:** No caso de erro, o sistema de arquivos é novamente montado em modo somente leitura.

**errors=panic:** A máquina pára no caso de erro no sistema de arquivos.

```
reiserfsck -rebuild-sb
```

Neste caso, o superbloco provavelmente está corrompido, de modo que o sistema de arquivos não pode ser acessado. O usuário pode restaurar o superbloco com *rebuild-sb*, se na partição realmente houver um sistema de arquivos Reiser.

## Expandindo o ReiserFS

O programa *resize\_reiserfs* ainda está no estágio beta e por isso deve ser usado com cautela. A ferramenta é capaz de

aumentar ou reduzir o tamanho do sistema de arquivos, que não deve estar montado durante a operação. O comando é bastante simples de usar:

```
resize_reiserfs -s -2G /dev/hdb2
```

Com isso diminui-se o tamanho do sistema de arquivos em */dev/hdb2* em 2 GBytes. Para aumentar o sistema de arquivos em 2 GB, basta substituir *-2G* por *+2G*. Infelizmente a coisa não é assim tão simples: além de executar a

## Sysctls do XFS

• *fs.xfs.stats\_clear* (Min: 0; Padrão: 0; Max: 1): O valor 1 apaga as estatísticas em `/proc/fs/xfs/stat` e `/proc/fs/xfs/stat`.

• *fs.xfs.sync\_interval* (Min: HZ, Padrão: 30\*HZ; Max: 60\*HZ): Estabelece o intervalo em que o `xfsyncd` escreve os metadados no disco.

• *fs.xfs.error\_level* (Min: 0; Padrão: 3; Max: 11): Define a extensão dos relatórios de erro. Gera informações mais ou menos detalhadas e rastreia shutdowns do sistema de arquivos. Os seguintes valores de nível de erro estão definidos: OFF: 0; LOW: 1, HIGH: 5.

• *fs.xfs.panic\_mask* (Min: 0; Padrão: 0; Max: 127): Opção para depuração de erros. Em casos específicos, a função `BUG()` é chamada.

• *fs.xfs.irix\_symlink\_mode* (Min: 0; Padrão: 0; Max: 1): Controla se os links simbólicos são criados com o modo padrão (0777) ou se o modo depende do padrão especificado com `umask` (modo IRIX).

• *fs.xfs.irix\_sgid\_inherit* (Min: 0; Padrão: 0;

Max: 1): Verifica os arquivos que o usuário coloca em diretórios SGID. A opção faz com que o bit ISGID seja apagado, caso o ID de grupo de arquivos novos não coincida com os IDs de grupo efetivos ou com um dos IDs adicionais dos diretórios pais.

• *fs.xfs.restrict\_chown* (Min: 0; Padrão: 0; Max: 1): Verifica se usuários não privilegiados podem usar o comando `chown` para transferir a propriedade sobre arquivos a outros usuários.

• *vm.pagebufstats\_clear* (Min: 0; Padrão: 0; Max: 1): O valor 1 apaga imediatamente todas as estatísticas em `/proc/fs/pagebuf/stat`.

• *vm.pagebufflush\_age* (Min: 1\*HZ; Padrão: 15\*HZ; Max: 300\*HZ): A idade a partir da qual o conteúdo dos buffers de metadados são escritos no dispositivo de bloco.

• *vm.pagebufflush\_int* (Min: HZ/2; Padrão: HZ; Max: 30\*HZ): O intervalo no qual a lista de buffers de metadados é verificada.

operação anterior, o usuário ainda tem que adaptar a partição ao novo tamanho do sistema de arquivos. No caso de um aumento, isso deve ser feito antes, no de uma redução, depois. O tamanho do journal, bem como o tamanho máximo de transação, podem ser alterados com o programa `reiserfstune`. Além disso, com este comando o usuário também pode mover o journal para um outro dispositivo de bloco.

O programa `debugreiserfs` auxilia na busca por erros. Chamado sem opções, ele retorna o superbloco do dispositivo indicado. Neste contexto, a opção `-p` é interessante: com ela o `debugreiserfs` encontra e extrai os metadados do sistema de arquivos para que a equipe de desenvolvedores do ReiserFS possa eliminar o erro.

```
debugreiserfs -p /dev/hda2/ | gzip -c > hda2.gz
```

Uma desvantagem para muitos usuários de Ext 2/3 que consideram uma migração para o ReiserFS é que, até o momento, não existe ferramenta de Dump/Restore para este sistema. As FAQs do ReiserFS recomendam usar o utilitário `tar` para fazer backups incrementais. Entretanto, usuários da distribuição SuSE-Pro ganham um sistema de backups completo via YaST2.

## XFS: para os exigentes

Juntamente com o Ext 3, o XFS dá a impressão de ser o sistema de arquivos mais completo. Ele não só disponibiliza numerosas opções, como também sua documentação é exemplar. É o único sistema de arquivos que permite colocar Sysctls no sistema `/proc` (ver Tabela 1). O tamanho de bloco é ajustável até 4 kB. Usuários esfomeados por performance são servidos pelo XFS com opções para diversos tamanhos de buffer. Destaques são o suporte a quotas no kernel e as ferramentas no pacote do `xfsprogs` [4]. Além disso, o pacote contém ferramentas de dump/restore próprias do XFS.

## Funcionalidade total!

Nos casos mais simples, o usuário formata uma partição com o comando:

```
mkfs.xfs /dev/sda1
```

Se o registro de metadados (o journal) deve residir em outra partição, entram em cena as seguintes opções:

```
mkfs.xfs -l logdev=/dev/sdb2, size=10000b /dev/sda1
```

Com isto a primeira partição do primeiro disco SCSI é formatada, e o journal é colocado na segunda partição do segundo disco.

O XFS se comporta de maneira bastante flexível no quesito tamanho de bloco – mas infelizmente não no Linux. Blocos de 512 bytes a 64 kB são possíveis, mas no Linux o tamanho do bloco pode ser, no máximo, do tamanho do Pagesize, que é limitado a 4 KBytes em sistemas x86. O usuário pode, entretanto, selecionar um tamanho menor.

As outras opções oferecidas pelo XFS estão agrupadas de acordo com funções para manipulação da área de dados do sistema de arquivos, de inodes, do journal e para ajuste do tempo real do sistema. Cada um dos grupos fornece dados de uma determinada área do disco rígido. Assim, pode-se ajustar o tamanho da fatia de sistemas RAID, ou indicar o número de inodes variáveis em porcentagem. Uma visão geral dos parâmetros do sistema é fornecida com a opção `-N`.

O XFS também pode ser expandido: o comando `xfs_growfs` aumenta a seção dos dados para o maior tamanho possível (com a opção `-d`). A opção `-D` permite ao usuário indicar o tamanho dos blocos. A seção de tempo real do sistema de arquivos pode ser aumentada com a opção `-r`. Como no caso da seção de dados, um tamanho determinado pode ser fornecido com a opção `-R`.

Geralmente se utiliza o aumento do tamanho do sistema de arquivos em combinação com o LVM (Logical Volume Manager), simplesmente acrescentando um outro disco ao sistema, e adicionando este disco ao grupo de volumes.

Os especialistas também têm com o que se divertir: com `xfs_db` é possível manipular certos blocos, lendo-os ou escrevendo sobre eles. A opção `blockuse` fornece o nome do arquivo que utiliza um determinado bloco. A enorme lista de opções do `xfs_db` mostra que, graças ao XFS, o Linux já tem condições de concorrer de igual para igual com os sistemas UNIX tradicionais. ■

## INFORMAÇÕES

- [1] Ezfsprogs: <http://ezfsprogs.sourceforge.net/ext2.html>
- [2] Jfsutils: <http://www-124.ibm.com/jfs/>
- [3] Reiserfsprogs: <http://www.namesys.com/download.html>
- [4] Xfsprogs: <http://oss.sgi.com/projects/xfs/download.html>