

Fundamentos dos sistemas de arquivos com journaling

Bloco da Alegria!

Um sistema de arquivos pode ser representado como uma camada de minúsculos blocos sobre a superfície de um disco. No início esta camada está organizada, porque todos os dados estão dispostos de modo limpo, uns depois dos outros, em blocos contíguos. Entretanto, a ordem se perde assim que os dados são postos em movimento (gravados e regravados), surgindo assim um efeito chamado fragmentação. Os novos sistemas de arquivos para Linux lidam melhor com esse processo do que o tradicional sistema de arquivos Ext 2. Em um teste comparativo detalhado entre o Ext 2 e o ReiserFS [1], o ReiserFS se comportou nitidamente melhor no tratamento da fragmentação de dados.

As longas pausas para o café, outrora necessárias quando o programa de verificação de integridade (*fsck*) examinava um sistema de arquivos Ext 2, foram abolidas. Os novos sistemas de arquivos para Linux registram as mudanças no sistema de arquivos em uma espécie de catálogo, o *journal* e por isso eles são chamados de sistemas de arquivos com *journaling*. Graças ao *journaling*, o programa *fsck* já terá reparado inconsistências no sistema de arquivo, enquanto o *fsck.ext2* ainda estiver verificando se existem efetivamente erros a eliminar. Motivo: o *fsck.journal*, utilizado pelo *fsck*, examina somente os dados anteriores à última entrada no *journal*. Com isso, o computador fica pronto para uso mais rapidamente e a disponibilidade do sistema aumenta.

Tamanho É documento.

A alta disponibilidade é importante porque o volume de dados cresce sem parar e, com ele, também a capacidade do hardware. Os fabricantes de discos rígidos já superaram a barreira dos 300 GB e há tempos os sistemas RAID já administram numerosos terabytes (TB). De qualquer modo, os projetistas de sistemas de arquivos pensam muito além do hardware disponível hoje e criam sistemas

Os quatro sistemas de arquivos com *journaling* disponíveis para o kernel 2.6 devem ser invisíveis e confiáveis. Ext 3, JFS, ReiserFS e XFS superam o Ext 2 em desempenho e fragmentam mais devagar. Além disso, graças ao *journal*, a reinitialização do sistema após um *crash* é muito mais rápida.

POR JÖRG REITTER



capazes de administrar eficientemente mais de um milhão de terabytes (um exabyte, 1 EB), conforme ilustra a Tabela 1. O XFS da SGI lidera, com um tamanho máximo teórico para o sistema de arquivos de 1,125 EB no Linux, um pouco à frente do ReiserFS, cujo limite está em 1 EB. A IBM segue em terceiro, com 32 PB (petabyte: 1 PB corresponde a 1024 TB). O Ext 3 está restrito às limitações impostas pelo kernel 2.6, com um tamanho máximo de 16 TB. A versão 2.4 pode administrar, no máximo, 2 TB.

Se o espaço em disco se tornar insuficiente, o usuário pode aumentar o tamanho dos sistemas de arquivos com o sistema ainda em funcionamento. Isto é especialmente interessante em conjunto com um volume lógico (*LVM - Logical Volume Manager*) ou com RAID (*Redundant Array of Inexpensive Disks*) via software. O artigo “Jogue seus dados”, na página 38 desta edição, descreve o uso das duas técnicas para aumentar, de forma quase ilimitada, o espaço em disco. O limite é o seu bolso.

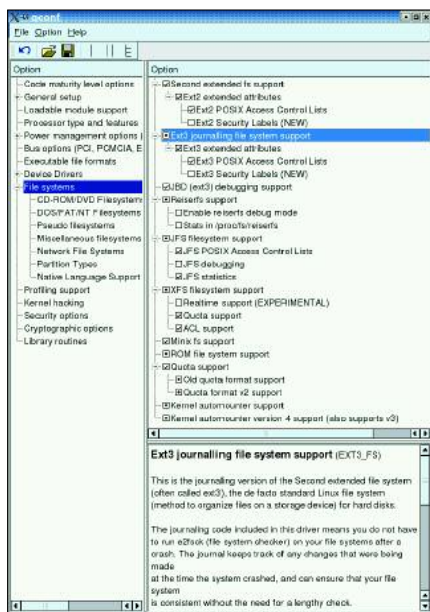


Figura 1: No kernel 2.6 há suporte para os sistemas de arquivos Ext 2/3, JFS, Reiserfs e XFS. As diferenças estão no suporte a ACLs, quotas e funções de debug.

Sistemas de arquivos no kernel 2.6

O kernel 2.6 contém drivers para sistemas de arquivos Ext 3, JFS, ReiserFS e XFS, com os quais o usuário pode realizar testes confortavelmente, sem precisar alterar e recompilar as fontes originais. O kernel não só dispõe destes drivers, como também de funções modernas como suporte a quotas, listas de controle de acesso (“Access Control Lists”, as famosas ACLs) e suporte à “debugging” (depuração de erros), conforme mostra a Figura 1. O que permanece é o fato de que o suporte ao sistema de arquivos da partição *root* precisa estar compilado no kernel, caso um RAM disk inicial (*initrd*) não seja utilizado. Naturalmente, o usuário ainda pode compilar e carregar os sistemas de arquivos como módulos em tempo de execução.

No que concerne à administração de direitos do sistema de arquivos, os administradores de rede têm apelado claramente, nos últimos tempos, cada vez mais às ACLs. Elas expandem os padrões habituais de direitos do usuário (“user”, o proprietário de um arquivo), do grupo de usuários e outros, de modo que o administrador pode facilmente, por exemplo, liberar o acesso a arquivos em um servidor Samba para usuários Windows remotos. Para o usuário de Windows, os direitos de acesso aos arquivos

aparecem como se ele acessasse um servidor Windows NT/2000, ao invés de um servidor Samba. Veja mais sobre a configuração de ACLs em [2] e [3].

Dividir e conquistar

Nos servidores de arquivos há sempre problemas de espaço. As quotas são uma solução eficaz para evitar que o espaço em disco do servidor acabe. O administrador determina, para cada usuário, uma fatia fixa do sistema de arquivos. Os sistemas de arquivos Ext 2/3 usam o suporte a quotas padrão do kernel 2.6, ao contrário do XFS, que oferece suporte próprio. O desenvolvimento do JFS e do ReiserFS, entretanto, ainda não chegou tão longe: no sistema de arquivos da IBM, o suporte a quotas está no cronograma para versões futuras. O ReiserFS só terá suporte a quotas a partir de sua versão 4, mas quem precisar dele hoje pode aplicar um patch no kernel 2.4.

Com os programas utilitários temos um quadro similar: JFS e ReiserFS não trazem programas para “dump” e “restore”. O YaST2, do SuSE Linux, fornece, entretanto, uma interface de backup para o ReiserFS – contudo sem suporte a backups incrementais. Deste modo, os usuários devem obrigatoriamente se habituar ao uso dos aplicativos *tar* ou *cpio*. Os usuários do Ext 3 simplesmente utilizam os programas habituais do Ext 2 [4]. A equipe de desenvolvimento de sistemas de código aberto da SGI também criou um utilitário para backup próprio, disponível no pacote *xfsplogs* [5]. Naturalmente existem outros programas que também podem fazer backups incrementais e a área de TI das empresas devem sempre prever essa necessidade [6].

Rapidinho com journaling

Sistemas de arquivos com journaling contribuem muito para a alta disponibilidade no datacenter. Sistemas assim equipados recuperam-se e retornam à operação muito mais rapidamente após um crash do que sistemas de arquivos sem journal, como o Ext 2. Uma reinicialização não planejada do computador implica em risco de inconsistência de dados no sistema de arquivos, pois arquivos que estavam em uso naquele momento não foram corretamente fechados. Durante a reinicialização, o programa *fsck.ext2* precisa verificar todos os arquivos quanto à consistência, o que pode levar horas em grandes sistemas de arquivos. Isso já não acontece com sistemas de arquivos equipados com journaling. Durante a reinicialização, o programa *fsck.journal* examina somente os arquivos posteriores à última entrada no journal, de modo que o sistema torna-se operacional novamente em segundos (ao invés de horas).

Mas não é somente após um crash que sistemas de arquivos com journaling são superiores. No que tange à eficiência organizacional em nível mais baixo, eles superam o Ext 2 de longe: seu esquema organizacional se baseia em determinados algoritmos de busca e classificação, chamados “B-Trees” (ver Quadro 1: Árvores Balanceadas).

Em contraste com uma lista desorganizada, como a utilizada pelo Ext 2, em um sistema com journaling as referências aos blocos individuais, que representam um arquivo, são organizadas na forma de uma árvore. Esta estrutura é tão otimizada, que um processo precisa esperar no máximo a metade do tempo

Fragmentação: perda programada de performance

Um arquivo com tamanho de 10 kB está num sistema de arquivos com blocos com tamanho de 4 kB. O arquivo não cabe, portanto, num único bloco e o sistema de arquivos o distribui em três. A princípio, esses blocos estão em seqüência, um atrás do outro. Entretanto, em um sistema de arquivos muita coisa acontece: os usuários excluem e acrescentam novos arquivos e relatórios dos processos também aumentam seus registros. Com o alvoroço desta movimentação de dados, formam-se cada vez mais ilhas de blocos onde antes havia um sólido bloco de arquivos. O sistema de arquivos se fragmenta continuamente, o que faz com que o desempenho vá caindo pouco a pouco (ver também [1]).

Um programa de desfragmentação seria ideal para restabelecer a ordem. De qualquer maneira, a desfragmentação, com as dimensões dos discos atuais, só deve ser feita se houver muito tempo disponível para isso. Num ambiente de produção, é melhor formatar o disco novamente e restaurar o backup dos dados de volta para ele depois. Pouco importa se o sistema de arquivos tem 200 GB ou 20 TB de dados – cedo ou tarde a morte o atingirá.

que esperaria normalmente, até que o sistema de arquivos encontre os blocos pertencentes ao arquivo procurado.

Já que, vistas do ponto de vista da raiz, as ramificações da árvore apresentam sempre a mesma profundidade, o termo B-Trees transforma-se, no jargão da literatura técnica, em árvores balanceadas. Cada um dos três sistemas de arquivos com journaling que utilizam uma estrutura de arquivos baseada em árvores balanceadas usa uma variante diferente: uns são, assim, mais eficientes no tratamento de vários arquivos pequenos em um diretório, outros, no tratamento de arquivos muito grandes.

Um gargalo apresentado por todos os sistemas, com exceção do Ext 3, é o VFS (“Virtual Filesystem Switch”). Sua arquitetura foi desenvolvida em função de sistemas de arquivos baseado em inodes. Entretanto, o JFS, o ReiserFS e o XFS usam B-Trees ao invés de inodes. As B-Trees aceleram o procedimento de busca, mas criam também um problema: um único erro em sua estrutura já pode levar à perda de dados. Além disso, exportar tais sistemas de arquivos via NFS é uma operação problemática e, além disso, o JFS, o Reiserfs e o XFS não funcionam com as quotas padrão do kernel.

Ext 3 ou Ext 2 com journaling

Quem usa o Ext 3 [7] ou prefere o Ext 2 não assume riscos. Não somente a possibilidade de transformar um sistema Ext 2 num Ext 3 através do programa *tune2fs* é uma vantagem importante, mas também os programas de dump e restore são os mesmos. Em comparação com o Ext 2, o Ext 3 ganhou só a função de journaling, com três opções de configuração: *journal*, *writeback* e *ordered*.

Com a opção *ordered*, se um arquivo e seus metadados correspondentes são modificados, estes dados são, antes, gravados (“flush”) no disco. Este é o comportamento padrão. No modo *writeback* somente os metadados são gravados no journal – o arquivo é modificado somente durante a sincronização do sistema de arquivos (“sync call”). O modo mais seguro, e mais lento, é o modo *journal*, que protocola as modificações nos metadados e nos arquivos.

A grande desvantagem do Ext 3 é a implementação de diretórios, feita através de uma lista encadeada. Todos os

Quadro 1: Árvores Balanceadas

Com as árvores balanceadas, os sistemas de arquivos para Linux utilizam um conceito que é aplicado há mais de 30 anos às bases de dados. A distribuição regular (balanceada) dos dados pelo sistema de arquivos com journaling resulta num esquema de organização mais eficiente que a técnica em uso pelo sistema de arquivos Ext 2. O Ext 2 usa listas desordenadas, que são ineficientes para busca e exclusão de dados, uma vez que, no pior dos casos, o diretório inteiro precisa ser varrido em busca do arquivo.

Com as B-Trees, um esquema de ramificações entra na estrutura organizacional dos dados. Ramificações que provêm da mesma raiz têm sempre a mesma profundidade, de forma que o equilíbrio entre as operações realizadas na árvore pode ser garantido. Desta forma, a eficiência da busca em um grande volume de dados aumenta, uma vez que nunca é necessário varrer um diretório inteiro. Os dados são organizados numa estrutura de árvore, onde as folhas armazenam os dados, as ramificações (nós) contêm referências ao próximo nível da árvore ou, no final, às folhas.

Algoritmos otimizados levam à velocidade máxima

A B-Tree padrão armazena dados e chaves nos chamados nós internos e nos nós das folhas. Nó é um termo geral que designa um elemento da árvore, e pode significar a raiz, uma ramificação ou uma folha. O número máximo de nós que levam ao nó mais baixo (a folha) é denominado altura de uma árvore parcial (todos os elementos abaixo de um nó). A árvore é dita balanceada (ou em equilíbrio) se o número mínimo de nós existentes no caminho que leva da raiz de cada árvore parcial até uma folha diferir no máximo em 1 do número máximo de qualquer outro caminho. Uma explicação detalhada sobre as B-Trees pode ser encontrada no site da Linux-Fibel (www.linuxfibel.de/filesys.htm#supp2). O documento “Sorting and searching algorithms”, que se encontra em <http://epaperpress.com/sortsearch/download/sortsearch.pdf>, esclarece como funciona a implementação na linguagem de programação C.

outros sistemas de arquivos com journaling se baseiam em árvores balanceadas (ver Quadro 1: Árvores Balanceadas). Listas encadeadas apresentam um desempenho bem pior. A performance do Ext 3 piora nitidamente com o aumento no volume de arquivos, e quando se trata de encontrar arquivos. O Ext 3 também é fraco no quesito fragmentação externa. Uma análise deste comportamento por um longo período pode ser encontrada em [1].

Comparado com outros sistemas de arquivos com journaling, o Ext3 também fica pra trás em casos como arquivos muito grandes (> 1 TB), partições muito grandes ou muitos arquivos em um diretório. A razão para isso é que as estruturas internas apresentam muito poucos bits para lidar bem com tais casos.

JFS: subnutrido

O JFS [9] da IBM visa servidores de bancos de dados, servidores de arquivos e qualquer outra aplicação com grande necessidade de espaço em disco rígido. Além disso, o desenvolvimento da IBM valoriza principalmente a segurança dos dados e a escalabilidade do sistema. Apesar disso, o desenvolvimento do JFS para Linux está um pouco atrás do de

seus concorrentes. A lista de coisas a ser feitas é a mais longa entre eles e o sistema é o que traz menos opções. Mesmo assim, uma partição formatada em JFS já pode ter seu tamanho aumentado durante a operação regular do sistema, que também suporta as ACLs Posix.

O usuário pode analisar problemas no sistema de arquivos ativando *JFS debugging* na configuração do kernel. As mensagens de erro geradas serão, assim, escritas no *syslog*. A IBM também pensou nos administradores que gostam de estatísticas: com a opção adequada ativada na configuração do kernel, um conjunto de estatísticas a respeito da utilização do sistema pode ser encontrado em */proc/fs/jfs/*.

A organização de diretórios no JFS foi muito bem projetada. Pequenos diretórios, com no máximo oito arquivos, são gravados pelo sistema diretamente no inode. Todos os diretórios com mais arquivos são organizados em uma B-Tree. Como é um sistema de arquivos de 64 bits, O JFS é predestinado para a utilização com arquivos e partições “gigantes”. As ferramentas de administração para o usuário estão reunidas no pacote *jfsutils*. Para os kernel 2.6 o usuário vai precisar no mínimo da versão 1.0.14.

ReiserFS: preferência por “arquivinhos”

Há tempos se diz que a quarta versão do sistema de arquivos de Hans Reiser [8] logo fará a alegria dos servidores FTP deste mundo. Vamos reservar para uma futura edição da Linux Magazine uma análise do desempenho do recém-lançado Reiser4, como a nova versão do sistema de arquivos é chamada. No presente artigo examinamos com cuidado os recursos das últimas atualizações da versão 3, mais especificamente o ReiserFS 3.6, que se encontra no kernel Linux atual. Quem quiser se manter informado sobre as novidades da versão 4 deve se preparar para participar de uma lista de discussão bem ativa.

O ponto alto do ReiserFS é a sua excelente performance quando se encontram no sistema muitos arquivos pequenos. Pequeno aqui significa menores que 1 kB. Arquivos com menos que 1024 bytes são muito comuns em servidores de “news”, razão pela qual administradores de sistemas recomendam o uso do ReiserFS para este tipo de serviço. Por outro lado, o sistema não apresenta um desempenho tão bom quando tem que acrescentar e excluir muitos arquivos, como acontece, no diretório */tmp*.

O ReiserFS manipula arquivos com grande eficiência no que diz respeito a aproveitamento de espaço em disco: ele não os aloca em blocos de tamanho fixo de 4 kB, como é tradição, mas em blocos cujo tamanho é exatamente o necessário. Adicionalmente, ele comprime a parte final de vários arquivos que não preenchem um bloco inteiro, e as grava juntas em um bloco separado (técnica chamada “tail packing”).

No quesito escalabilidade, o ReiserFS vai muito bem, obrigado: teoricamente, o sistema de arquivos pode ter até, no máximo, 1 EB (1 Exabyte), bem como pode-se trabalhar nele com arquivos da mesma ordem de grandeza.

XFS: o futuro é agora

O XFS, da SGI, [10] está nitidamente mais desenvolvido que o JFS ou o ReiserFS. Ele dispõe de várias opções para ajuste de desempenho, a documentação é muito boa e, para cada ferramenta contida no pacote *xfsprogs*, existe uma página de manual (*man page*) correspondente. A implementação de código aberto feita pela SGI para o kernel 2.6.0 já contempla inclusive ACLs (*Access Control Lists*). Além disso, o XFS é o único sistema de arquivos a oferecer um suporte próprio para quotas. O XFS trata as informações sobre quotas como metadados e as integra ao journal. Assim, uma alta consistência para as quotas fica garantida. Por último, o formato das quotas é compatível com o utilizado no IRIX, sistema operacional Unix da própria SGI, o que permite a migração de dados sem a necessidade de conversão do sistema de arquivos.

O XFS, tal como o JFS, é um sistema de arquivos de 64 bits, estando assim bem equipado para abrigar arquivos de proporções monstruosas. Teoricamente, o sistema de arquivos pode ter, no IRIX, até 18 EB (18 Exabytes, ou seja, 19.327 352.832 Gigabytes) e conter, por exemplo, apenas dois arquivos de 9 Exabytes cada. Não é de se admirar que ele seja o único sistema de arquivos equipado com journaling capaz de oferecer redimensionamento em tempo de execução.

Conclusões

À exceção do Ext 3, os sistemas de arquivos com journaling são tecnicamente bem semelhantes. Somente no atual estado de desenvolvimento de cada um deles há diferenças. Como um sistema de arquivos Ext 2 pode ser facilmente convertido em um Ext 3, e este, além disso, é compatível com o primeiro, o Ext 3 deverá ser a primeira escolha para muitos. O que falta ao sistema em desempenho, ele compensa em robustez. Quem, por outro lado, estiver procurando por um sistema de arquivos com escalabilidade enorme e alta performance, não tem como escapar do XFS. Uma gama completa de aplicações e sua boa documentação são vantagens adicionais do sistema de arquivos desenvolvido pela SGI. A utilização do ReiserFS 3.6 está indicada para casos em que muitos arquivos pequenos devem ser manipulados pelo sistema. Vamos ver o que a versão 4 do sistema nos reserva. O JFS ainda não chegou ao mesmo estágio que o dos outros sistemas de arquivos, de modo que o seu uso ainda não é recomendado em ambientes de produção. ■

Recursos dos sistemas de arquivos

Recurso	Ext3	JFS	ReiserFS	XFS
Journaling:Dados/Metadados	s/s	n/s	n/s	n/s
Journal Replay Kernel/Aplicações	s/s	n/s	s/n	s/n
Journal Interno/Externo	s/s	s/s	s/n	s/s
Diretórios baseados em B-Trees	n	s	s	s
Mapa dinâmico de alocação de inodes	n	s	s	s
Atributos estendidos/ACLs Posix	s/s	n/s	n/n	s/s
Quotas	s	Patch	Patch	s
Aumento/redução offline	s/s	n/n	s/s	s/n
Aumento/redução online	n/n	s/n	s/n	n/
Tamanho de bloco implementado (em kB)	1,2,4	4	4	4
Tamanho máximo do sistema de arquivos	16 TB	32 PB	1 EB	1,125 EB
Tamanho máximo dos arquivos	2 TB	4 PB	1 EB	562,5 PB

INFORMAÇÕES

- [1] Fragmentação: <http://www.informatik.uni-frankfurt.de/~loizides/reiserfs/thesis.html>
- [2] Curso sobre ACLs: <http://acl.bestbits.at/>
- [3] ACLs no SuSE: http://portal.suse.de/sdb/de/2002/10/81_ad.html
- [4] Dump/Restore para o Ext 2/3: <http://sourceforge.net/projects/dump/>
- [5] Xfsprogs: <ftp://oss.sgi.com/projects/xfs/>
- [6] Software de backup: <http://www.linuxmafia.com/faq/Admin/tape-backup.html>
- [7] Third Extended Filesystem (Ext3): <http://www.zip.com.au/~akpm/linux/ext3/>
- [8] ReiserFS: <http://www.namesys.com/>
- [9] JFS: <http://oss.software.ibm.com/jfs/>
- [10] XFS: <http://oss.sgi.com/projects/xfs/>
- [11] White Paper da Red Hat sobre o Ext 3: <http://www.redhat.com/support/wpapers/redhat/ext3/index.html>
- [12] Journaling Filesystems: http://en.wikipedia.org/wiki/Journaling_filesystem
- [13] Introdução ao Reiser4: <http://www.kuroshin.org/story/2003/8/9/172159/7912>