O poder dos daemons do Conde Bacula

Backup Imortal

Quando seus backups ficam complexos demais para um simples script, o onipotente Bacula, livre e gratuito, pode ser a resposta.

POR JENS-CHRISTOPH BRENDEL

As políticas de backup variam de tamanho, preço, cor, credo e inclinação política. As mais mequetrefes baseiam-se em scripts simples talhados para cobrir as hecatombes, usando para isso as ferramentas nativas do sistema operacional (como *tar*, *dd*, *cpio*). Essa abordagem é suficiente para backups locais com pouco volume de dados e uma meia-dúzia de clientes.

Políticas um pouco mais parrudas requerem técnicas mas requintadas. Utilitários como o *rsync* e o *Amanda* são eficientes para um grande número de situações, mas essas ferramentas normalmente exigem o desenvolvimento de scripts sofisticados (e complicados) e possuem algumas limitações – nem sempre visíveis num primeiro momento – de tempo, sincronização de dados, volume e reconhecimento de hardware.

As ferramentas de porte corporativo acabam com todas essas restrições. Como toda rosa tem seus espinhos, os fabricantes dessas ferramentas esotéricas também cobram preços do outro mundo. Uma exceção a essa regra é o *Bacula* [1], um sistema de backups gratuito e livre que oferece um cardápio variado de recursos, a maioria deles também presentes nos menus de produtos muito mais caros.

O Bacula não é um sistema monolítico. Pelo contrário, é um conjunto de vários *daemons* precedidos de uma interface com o usuário. Cada daemon possui uma função específica e usa a rede para se comunicar com os demais. Com isso podemos dividir a carga de trabalho: o trabalho árduo – ou seja, ler e gravar dados – é feito por uma equipe de daemons nos computadores clientes (os que possuem dados a gravar) e nos servidores de backup (os que guardarão os dados para a posteridade). O controle de tudo isso é feito na estação de trabalho do administrador; um servidor de banco de dados controla as contas dos usuários. Cada um desses elementos pode estar em uma máquina diferente, mas nada impede que se agrupe mais de uma função em um determinado sistema ou mesmo que se use um único computador para tudo. Essa flexibilidade torna o sistema facilmente escalável (**figura 1**).

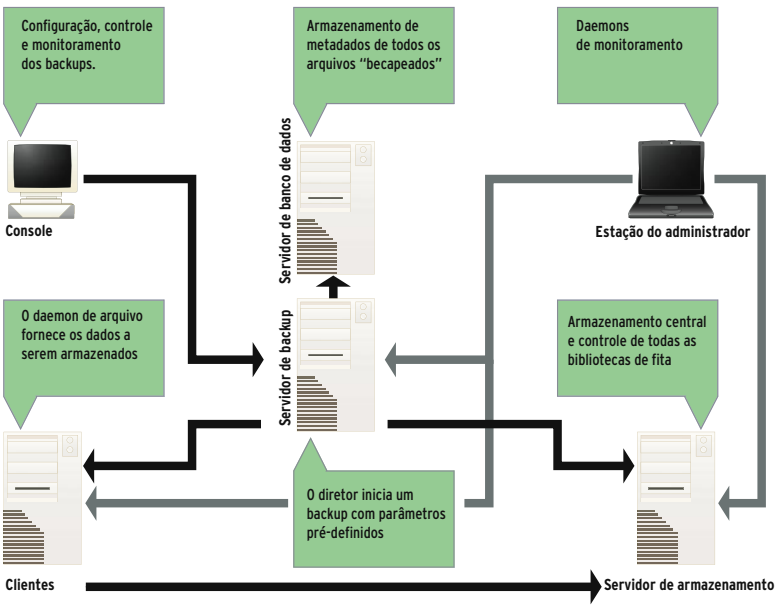

Figura 1: Dividir para conquistar – o *Bacula* distribui o trabalho pela rede, mas mantém o repositório de dados em um ponto central.

Liderança central

O patrão de toda a equipe de daemons é chamado, apropriadamente, de *diretor* (*director*). O diretor sabe o que deve ser armazenado e onde. Sabe também onde estão os dados do usuário, caso este precise restaurá-los. Ele também conhece os cronogramas, clientes, silos de armazenagem e detalhes sobre as tarefas planejadas. Com todo esse conhecimento, o diretor pode distribuir as tarefas de seus subalternos, os daemons. O diretor – ele próprio um daemon – também possui a distinção de ser o único componente do castelo de Bacula a falar diretamente com um ser humano.

O diretor guarda os detalhes de configuração em um arquivo de texto puro (`bacula-dir.conf`) na forma de descrições, hierarquicamente estruturadas, dos recursos do sistema. A patente mais alta nessa hierarquia é ocupada pelo recurso de controle de tarefa (*job control*), que reúne as configurações de uma tarefa em especial – tipo de tarefa (backup, restauração, verificação ou administração), horário e duração da tarefa e o nível de profundidade da tarefa (para backups, por exemplo, temos os níveis *completo*, *incremental* e *diferencial*).

Para que as coisas fiquem mais fáceis, muitos dos detalhes são agrupados em "subrecursos", mais apropriadamente chamados de *diretivas*. Recursos comuns a diversas tarefas similares podem ser agrupados em *JobDefs* (Definições de Trabalhos) para formarem classes, que outras descrições de tarefas podem referenciar. Com esse "reuso de código", os arquivos de configuração podem ser simplificados e encurtados.

Por exemplo, o tipo de recurso *Schedule* (cronograma) define agendamentos que executam tarefas em intervalos específicos de tempo, sendo bastante flexíveis no tocante a tipos de cronograma. Já o recurso *FileSet* (conjunto de arquivos) lista diretórios e arquivos que você planeja guardar. Os diretórios são manipulados recursivamente, o que quer dizer que "`/`" resultará no tipo mais simples de backup, o de todo o disco – embora na prática o bom administrador prefira excluir alguns diretórios como o `/var` e o `/tmp` ou arquivos ocultos como `.journal` e `.fsck`.

O backup só guardará itens em sistemas de arquivos diferentes (i.e. partições diferentes, volumes em outras máquinas acessados pela rede, mídias removíveis) se houver uma ordem expressa para tal. Como padrão, apenas arquivos no mesmo sistema são salvos para evitar o perigo de entrar em loops infinitos ou inadvertidamente salvar servidores inteiros. Se quiser continuar usando essa medida de segurança e mesmo assim salvaguardar vários sistemas de arquivos diferentes, é necessário indicar explicitamente cada um dos sistemas.

Obviamente, o Bacula permite tarefas muito mais complexas que isso. Por exemplo, pode-se referenciar uma lista de arquivos externa, usar expressões em shell ("coringas") para escolher arquivos de acordo com um critério ou mesmo empregar scripts que geram listas de backup em tempo real. Como o uso de expressões em shell significa escrever tudo em uma única linha – e, portanto, todos os caracteres fora do padrão devem ser cuidadosamente "escapados" – os scripts são a opção mais viável.

Imagine, por exemplo, que por algum motivo estranho você queira guardar todos os arquivos de configuração que estão dentro do diretório `/etc` e todos os arquivos e sub-diretórios ocultos que estão dentro do diretório pessoal do usuário `jcb`. O mini-script a seguir deve dar conta do recado:

```
#!/bin/sh
find /home/jcb -maxdepth 1 -name ".*"
find /etc -name "*.conf"
```

O *FileSet* que gruda tudo isso seria:

```
FileSet {
name = "ConfigSet"
include {
Options {
signature = MD5
}
File = "|/etc/bacula/confbackup.sh"
}
}
```

Além de usar arquivos, listas e scripts, os administradores podem também especificar dispositivos brutos (*raw devices*) como fontes de dados (embora possam ser montados apenas para leitura). Por fim, o backup pode até mesmo ler dados de FIFOs (ou *named pipes*, como queira), possibilitando conectar o backup a um aplicativo que esteja rodando naquele momento. Entretanto, o nível fora do comum de flexibilidade no tocante a fontes de dados tem seu preço: selecionar essas fontes é dramaticamente menos intuitivo do que simplesmente deixar o administrador selecionar os arquivos em uma interface gráfica. Uma combinação das duas abordagens seria o ideal. Infelizmente nada é perfeito...

Bancos de inclusão

Outra diretiva de configuração define *bancos de backup* (*backup pools*), cada um possuindo uma determinada quantidade de volumes em fita. Essa diretiva, sozinha, coloca o Bacula

num patamar muito mais alto do que as outras soluções livres, a maioria muito simples. Um banco de backup agrupa logicamente uma certa quantidade de fitas, permitindo que um único backup possa se estender muito além da capacidade de uma única fita. Quando a tarefa de backup encontra um aviso de fim de fita, o Bacula continua a tarefa na próxima fita disponível que esteja dentro do mesmo banco de backup. Com essa diretiva, além da vantagem óbvia do volume muito maior de dados num único backup, podemos reciclar as fitas mais antigas do grupo depois de um período configurável de tempo.

Os recursos dos bancos de backup são controlados por algumas diretivas de configuração – por exemplo, o tempo de espera para reuso da mídia ou o número máximo de vezes que uma dada fita pode ser reutilizada. As configurações se aplicam a todas as fitas no banco, o que é bom; os administradores não precisam mais definir as preferências de cada fita no grupo – embora isso ainda possa ser feito, se desejado.

Associar fisicamente as fitas a um determinado banco de backup pode ajudar a organizar a bagunça nos armários de fita. O programa permite organizá-las por tipo de utilização, impedindo que sejam misturadas ou mesmo apagadas por acidente. Isso é especialmente importante quando você tem uma fita com um backup completo e, inadvertidamente, grava um backup incremental por cima. Sem o backup completo, qualquer incremento é inútil. Também é possível definir bancos para clientes individuais, dias da semana, grupos de máquinas e assim por diante.

Para a troca automática de fitas, o Bacula supõe que você possua uma fitoteca robotizada. O programa reconhece uma grande variedade de robôs, também conhecidos como autocarregadores, *autochangers* ou *autoloaders* com drives do tipo DAT, VXA2, DLT, LTO e AIT.

A ferramenta *Mtx* [2], usada pelo Bacula para controlar fitotecas, pode até mesmo ler códigos de barra, o que permite a robôs com leitores laser identificar uma fita sem que seja necessário colocá-la no drive para ler seu conteúdo. Em alguns casos – por exemplo, quando as fitas houverem sido classificadas manualmente na fitoteca – deve-se fazer uma reclassificação da localização das mídias. Se isso acontecer com você, acredite: agradecerá aos céus o fato de seu chefe ter liberado verba para a aquisição do sistema de reconhecimento de código de barras.

Catálogos

Sempre que o Bacula coloca um arquivo em uma fita, guarda também os detalhes desse arquivo – tamanho, atributos, assinatura, data da última modificação, data e local do backup – em um banco de dados conhecido como "o catálogo". Esse é o terceiro recurso principal do Bacula e o coloca num patamar mais alto do que os outros sistemas de backup, pois permite que arquivos individuais sejam restaurados sem que a fita (ou grupo de fitas) tenha que ser lida completamente. Os arquivos que desejar restaurar podem ser selecionados pela simples menção de seus metadados, que incluem a posição do arquivo na fita. Não há necessidade de ler a fita de cabo a rabo; em vez disso, o Bacula pode avançar a fita até onde

Uma visita ao castelo do Conde Bacula

No início do ano de 2004, um provedor de serviços de Internet, com sede em Stuttgart, Alemanha, estava querendo substituir seu sistema de backup, ligeiramente ancião. O Bacula era um dos competidores, ombro-a-ombro com inúmeras soluções comerciais.

O que convenceu o provedor, além do fato de o Bacula significar a economia de rios de dinheiro em licenças, foi o fato de que o programa é livre e portanto independente da política comercial e industrial de qualquer fabricante – evitando assim o famoso "abraço amigo de tamanduá". O provedor também procurava por soluções que se integrassem com seus sistemas de cobrança e bilhetagem, além de permitir a centralização das configurações.

O provedor decidiu implementar um piloto para testar o sistema numa situação real – e a situação real de um provedor de Internet é algo assustador! Na fase piloto, 32 máquinas com FreeBSD foram alvo de backup durante três meses, com o Bacula rodando em paralelo com as soluções existentes (eram mais de uma).

Depois do teste inicial com robôs de fita ter sido um sucesso, o provedor optou por uma combinação heterogênea de mídias: uma unidade de fita LTO 1 e alguns discos rígidos. Um ciclo de backup de sete dias (um backup completo mais seis incrementais) e um período de retenção de 4 semanas foi estabelecido. Os administradores puderam, depois, afinar o ciclo baseados nas informações do banco de dados do catálogo.

Os 32 sistemas do teste multiplexaram os backups em grupos de 10 a 20 fluxos paralelos de dados. O parâmetro de configuração `Maximum Concurrent Jobs` foi mexido para que isso fosse possível – e, no final, o efeito foi positivo em backups incrementais e diferenciais com respeito ao tempo requerido para completar cada tarefa, bem como na carga (tempo de CPU, acesso a disco e tráfego de rede) imposta a cada um dos sistemas. Sob condições reais, o sistema levou 19 horas para criar um backup completo de 450 gigabytes de dados, 90 minutos para um backup diferencial e apenas 40 minutos para um backup incremental.

O MySQL usado originalmente mostrou sinais evidentes de gargalos de desempenho em testes de longa duração, levando-o a ser substituído pelo PostgreSQL, o que melhorou absurdamente o desempenho das tarefas de restauração e reciclagem de fitas.

A configuração otimizada foi testada em operações do dia-a-dia por um período de várias semanas. Depois de completar esse teste final, a conclusão a que o provedor chegou foi que o Bacula era capaz, com galhardia, de preencher os requisitos necessários para o trabalho. Como não havia nada que impedisse o provedor de instalar o Bacula em todo o seu *data center*, assim foi feito e todos se regozijaram.

o arquivo está – ou, pelo menos, até o início da tarefa de backup. Adicionalmente, o catálogo armazena o histórico de todos os backups.

O Bacula pode usar qualquer banco de dados SQL para sua administração. O pacote inclui scripts de configuração para *PostgreSQL*, *MySQL* e *SQLite*. Como são bancos SQL bastante populares, o administrador pode fazer cópias de segurança das tabelas em uso e, se o que já é ruim ficar danado, fazer cirurgias "à mão" para corrigir alguma coisa. Um catálogo extraviado ou inconsistente é um dos desastres mais catastróficos que podem acometer um dado backup. Para diminuir o efeito de um catálogo perdido, o Bacula possui scripts que guardam o catálogo em um arquivo texto enquanto o trabalho de backup está sendo feito. Se algo der errado, pelo menos a versão anterior pode ser restaurada facilmente.

Incidentalmente, você pode usar o catálogo do Bacula para improvisar um sistema de detecção de intrusos *a là Tripwire* ou *Aide*. Duas funções integradas, que você pode rodar independentemente das de backup e restauração, comparam os metadados armazenados com os do sistema de arquivos atual. Pode-se descobrir, dessa forma, alterações não autorizadas em arquivos.

Trabalho de equipe

É claro que um diretor não é nada sem uma equipe digna desse nome. No Bacula, o diretor dá ordens para dois grupos distintos de subalternos: um ou mais *daemons* de armazenamento (*storage daemons*) e inúmeros *daemons* de arquivo (*file daemons*). Estes últimos rodam nos clientes – ou seja, os computadores que têm coisas que devem ser guardadas

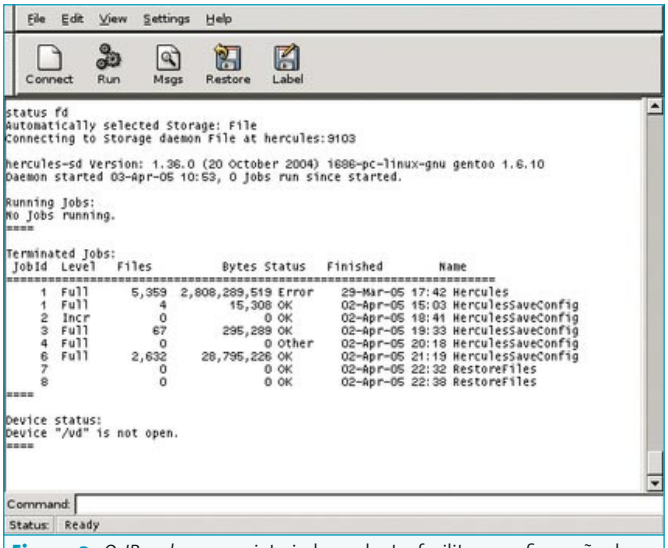

Figura 3: O *JBacula*, um projeto independente, facilita a configuração do daemon de diretório.

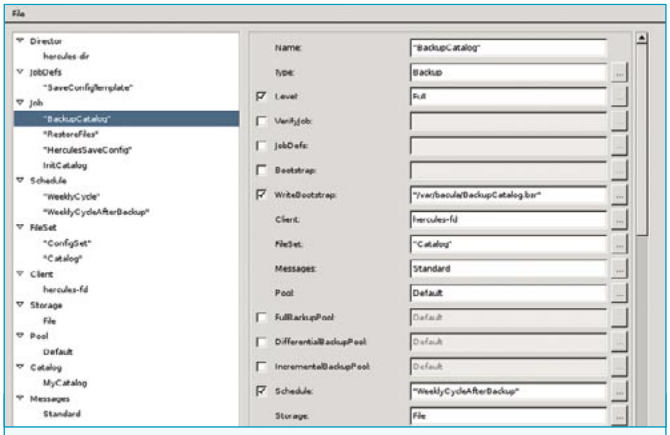

Figura 2: O *GConsole* não possui interface gráfica propriamente dita, mas oferece um console com uns poucos menus que não precisam de uma janela de terminal.

no backup – e usam a rede para enviar os dados ao servidor de armazenamento – que é o local onde o daemon de armazenamento está rodando e que controla o drive ou a fitoteca robotizada. Se necessário, o daemon de armazenamento pode também fazer backup em disco, o que pode ser uma solução de curto ou médio prazo para o armazenamento do último backup, especialmente se compararmos os preços cadentes dos discos rígidos e os cada vez mais altos das unidades (e mídias) de fita.

Daemons de arquivo estão disponíveis para Linux, a maioria dos sistemas operacionais "aparentados" com o Unix (por exemplo: Solaris, AIX, HPUX, FreeBSD, MacOS X) e todas as versões de Windows. O número de sistemas suportados evita que precisemos fazer gambiarras envolvendo NFS ou Samba para fazer backups de qualquer dos sistemas da rede – embora essas gambiarras, se existentes, funcionem muito bem com o Bacula!

Meia volta, volver!

O processo de restauração dos dados é o inverso do backup. Quando ordenado a isso pelo diretor, o daemon de armazenamento envia o arquivo ao daemon de arquivo, que por sua vez grava o arquivo solicitado no cliente. Os arquivos restaurados não são gravados em suas posições originais; em vez disso, uma árvore de diretórios completa contendo o arquivo é restaurada abaixo de um diretório especial. A configuração de cada tarefa de backup especifica qual é esse diretório. Obviamente, deve existir no cliente espaço livre o bastante para aceitar os arquivos restaurados. O padrão é `/tmp/bacula-restores`.

Você pode mudar esse comportamento especificando o raiz como diretório de restauração. Com isso, os arquivos são gravados em suas posições originais. Cuidado para não tomar

boi por bezerro: o Bacula não resolve conflitos que possam decorrer disso. Se um arquivo sendo restaurado já existir, o Bacula simplesmente vai escrever por cima sem dó nem piedade. Se o arquivo estiver protegido contra gravação, a restauração gerará um erro. Se não é exatamente isso o que você quer, o melhor é usar o método do diretório diferente. Confie em Alá, mas amarre o seu camelo.

Há mil e uma maneiras de se escolher quais arquivos do backup você quer restaurar. Todas elas acabam desembocando em um diretório virtual que mostra todos eles e a forma como estão organizados na fita. Pode-se navegar pela árvore de diretórios usando os comandos padrão do Unix (*cd*, *ls*, *pwd* e companhia limitada). Será preciso digitar comandos para marcar quais arquivos e diretórios devem ser restaurados – mais uma vez, uma interface gráfica em que se pudesse clicar nos arquivos desejados seria algo muito mais apropriado.

Como um serviço especial, o Bacula permite que se combine o último backup completo para um cliente e todos os incrementais subseqüentes. Adicionalmente, pode-se restringir a seleção a todos os arquivos em backup antes (ou depois) de uma determinada data e hora.

As versões atuais do Knoppix [3] incluem um daemon de arquivo e um console do Bacula, o que torna o sistema uma solução simples para recuperação de desastres, desde que você anote o particionamento de qualquer disco em que tenha guardado em backup – e também que armazene os arquivos de *boot* do Bacula em um local separado. Um CD

O futuro

O Bacula é, definitivamente, o sistema livre de backup que chega mais perto dos sistemas comerciais, em especial tratando da avidez gigantesca por dados dos ambientes corporativos. O conde Bacula é, indubitavelmente, talhado para uso em sistemas empresariais de todos os tamanhos. Como a chuva nem sempre é suficiente para salvar a lavoura, há ainda alguns itens que gostaríamos de ver em versões futuras:

- **Segurança**: no momento, não é possível fazer backups criptografados. Em outras palavras, um invasor poderia farejar (com um software para captura de tráfego ou *sniffer*) a rede local para acessar os dados do backup. O pânico com esse tipo de segurança não é histeria paranóica de administradores estressados, mas uma preocupação genuína em ambientes com dados importantes e sigilosos. O temor aumenta se o backup for efetuado por provedores de serviços, já que os dados trafegarão pela Internet ou por uma linha privada externa. Como paliativo, podemos usar um túnel SSH para criptografar a comunicação entre os daemons de arquivo e de armazenamento, bem como entre eles e o diretor. Em ambientes Windows seria preciso, no mínimo, integrar um anti-vírus atualizado. Os desenvolvedores do Bacula estão planejando as soluções para esse problema, mas nada de concreto foi implementado ainda.
- **Fitotecas grandes**: Embora múltiplos backups possam ser disparados simultaneamente, ainda é preciso melhorar de forma dramática a eficiência do processamento paralelo. Por exemplo, um daemon de arquivo não pode usar de multiplexação para enviar dados a múltiplos daemons de armazenamento. Isso melhoraria em muito o desempenho para grandes volumes de dados. Bancos de drives capazes de associar um determinado número de acionadores de fita a uma tarefa específica seriam algo bastante desejável. Se pudessem permitir que cada tarefa selecionasse o banco de drives que estivesse disponível (isto é, ocioso) no momento, melhor ainda. Tristemente, nenhuma das duas coisas é possível no momento. Também não há a possibilidade de atribuir automaticamente drives ociosos a tarefas pendentes. Sem esses recursos, uma fitoteca robotizada que disponha de múltiplas unidades de fita nunca estará com 100% de ocupação. Resultado: desperdício de recursos de hardware e perda de desempenho.
- **Interface gráfica**: a verdade é dura, mas o Bacula não possui *o* elemento primordial para que seja adotado em massa por administradores mundo afora – especialmente os oriundos de ambientes Windows. Embora algumas soluções tenham sido tentadas, não vão além de simples menus baseados em texto. Um gerenciador gráfico de arquivos para escolher o que guardar ou restaurar com um clique do mouse seria o mínimo aceitável. Um calendário para selecionar agendamentos seria bastante útil. Não há um assistente de configuração para ajudar os administradores nessa ingrata tarefa. Gurus experientes do Unix talvez não se importem muito com essas "frescuras", mas os administradores de hoje em dia vão procurar outra solução – e rirão bastante da sua cara – se você tentar impingir a eles uma solução baseada em linha de comando e scripts. Sem um ambiente gráfico no qual se possa configurar 100% (sem exceção) das opções do programa usando apenas o mouse – e um ambiente usável, por favor! – o Bacula está fadado a ser solenemente ignorado pelo público-alvo e entrará em extinção certa. O mesmo se pode dizer da ajuda online: além de existir (o que não é verdade, atualmente), ela deve ser completa e fácil de consultar.
- **Backup "a quente"**: Não há nenhum módulo para fazer backup de bancos de dados sem que seja preciso desativá-los antes. Também não é possível fazer backup de aplicativos e sistemas que usem arquivos e os travem (com *lockfiles*) para impedir o acesso por outros. O diretor contorna parte do problema permitindo que se rode scripts no cliente e no servidor antes e depois de iniciar alguma tarefa de backup. Nos scripts, poderíamos incluir comandos para encerrar a execução de tais sistemas e, depois, iniciá-los novamente. Como tanto o backup quanto a restauração podem usar FIFOs (*named pipes*) como suas fontes de dados (ou mesmo como destino), é possível manipular os dados indo ou vindo de aplicativos em execução sem que se tome um desvio por meio de um arquivo. Essa é uma alternativa interessante, mas de forma alguma pode substituir um verdadeiro e completo sistema de backup online.
- **Extras**: sistemas comerciais de backup oferecem a seus usuários um grande numero de programinhas extras, alguns bastante úteis, que o Bacula simplesmente não possui. Por exemplo, sistemas comerciais sempre incluem um utilitário para clonagem de mídias. Isso reduz ao mínimo o risco de erros irrecuperáveis nas fitas – claro, só se você as copiar na hora em que foram geradas. Também úteis e onipresentes são as ferramentas para recuperar e continuar sessões de backup interrompidas. É óbvio que o Bacula funciona sem esses programinhas, mas eles são um mimo ao qual os usuários desse tipo de sistema já estão acostumados. Sua ausência pode provocar a escolha de outros sistemas de backup mais pródigos.

de recuperação do Bacula, criado para reanimar o sistema depois de uma falha completa, não funcionará nos kernels mais recentes do Linux (2.6.x), mas uma versão atualizada está a caminho.

Designando responsabilidades

O acesso ao console do Bacula é governado pelas permissões de execução do usuário; o aplicativo não pede que os usuários se autentiquem e, portanto, não permite diferentes níveis de privilégio. Entretanto, é possível configurar variações no console para permitir apenas alguns tipos de tarefas e comandos, limitar determinados bancos de fitas e restringir drivers e fitotecas. Isso dá aos administradores uma forma improvisada (mas útil) de gerenciamento de usuários. O improviso não é granular o suficiente para permitir que cada usuário restaure seus próprios arquivos sem pedir permissão ao administrador, mas é possível criar grupos de usuários e delegar direitos administrativos a eles.

Em muitos casos, esperar que o usuário consiga restaurar os próprios arquivos sem uma interface do tipo "apontar-e-clicar" é algo temerário. Ferramentas como o *Wxconsole* e o *Gconsole* (**figura 2**) possuem alguns menus que evitam que o usuário tenha que saber de cor e digitar alguns comandos, mas ainda possuem uma linha de comando para as funções mais avançadas. O JBacula [4], escrito em Java, facilita um pouco a vida do usuário sem intimidade com o console (**figura 3**).

Conclusão

Os administradores que não têm medo da linha de comando vão encontrar no Bacula um sistema de backup utilíssimo e extremamente flexível, com muitos recursos profissionais. O Bacula é muito bem documentado e se integra facilmente com ambientes altamente heterogêneos. ■

Informações

[1] Castelo do Conde Bacula: www.bacula.org

[2] Mtx – Controle de fitotecas robotizadas (Frankenstein): mtx.badtux.net

[3] Knoppix e Bacula, dois monstros sagrados vivendo juntos: www.knopper.net/knoppix/index-en.html

[4] JBacula: jbacula.sourceforge.net