Notícias do Kernel

SquashFS e FUSE

A aceitação de sistemas de arquivos na árvore 2.6 parece estar ocorrendo caso-a-caso. O sistema de arquivos criptografado SquashFS de Phillip Lougher tem recebido opiniões irritadas de usuários, com números de desempenho que dão razão a elas. Greg Kroah-Hartman o tem mantido no kernel oficial do Gentoo no mesmo balaio que uma porção de outros patches com falhas não reparadas; ele tem insistido com Phillip para que este candidate o sistema de arquivo à inclusão na árvore oficial de Linus Torvalds.

O próprio Phillip está relutante, em parte por causa de uma persistente percepção da 2.6 como série "estável" e em parte porque não tem certeza se deve candidatar o código antes ou depois de alguns melhoramentos que quer adicionar. Mas, como o SquashFS influencia um único diretório e como o 2.6 está passando por um desenvolvimento maciço, além de haver pessoas como Greg clamando por inclusão, fica bastante claro que o SquashFS será aceito assim que Phillip decidir candidatá-lo.

O FUSE (*Filesystem in USErspace*), por outro lado, vem tentando entrar no 2.6, mas até agora tem sido rejeitado no campo técnico. Miklos Szeredi o candidatou recentemente, mas Linus Torvalds opôs-se fortemente ao patch, ao menos em sua forma atual.

Além de quaisquer detalhes de implementação que pudessem ser questionáveis, Linus acha que um sistema de arquivos no espaço do usuário é inerentemente bagunçado, pois tenta isolar a funcionalidade do resto do sistema operacional, embora na verdade devesse estar firmemente integrado a ele. Aliás, a respeito de um sistema de arquivos no espaço do usuário ele tem as mesmas objeções que tem para com um sistema operacional baseado em microkernel, como o Hurd. Ele acha que um sistema desses, ou um sistema operacional desses, nunca pode ser inteiramente bem-sucedido. No caso do FUSE, ele ofereceu algumas sugestões de mudanças que tornariam o patch mais aceitável e deixou implícito que poderia considerar o pach, mesmo em sua forma atual, se ele fosse submetido como diversos patches separados que fizessem coisas indiscutivelmente boas; assim, ainda há esperança para o FUSE.

Até o momento em que esta matéria foi escrita, porém, parece que os desenvolvedores do FUSE estão fazendo todo o possível para que ele seja incluído no 2.6 em curto ou médio prazo. ■

Software Suspend

O esforço para suspender um sistema ativo certamente já deu origem a muitas viravoltas e peripécias – e a corrida ainda não terminou. A versão 1 do software, desenvolvida principalmente por Pavel Machek, já está no kernel e funciona bem até certo ponto. A versão 2 do software, desenvolvida principalmente por Nigel Cunningham, não é realmente um salto de versão, mas uma reimplementação por um caminho um pouco diferente.

Todos parecem concordar que a versão 2 deve acabar por substituir a versão 1, mas Nigel está tendo que saltar através de muitos aros de fogo na jornada para a aceitação. Uma das preocupações de Nigel, e não a menor delas, é o fato de que Pavel, Christoph Hellwig e outros querem patches que transformem aos poucos a versão 1 na versão 2. Esse é o grande pesadelo de Nigel, já que as duas implementações são muito diferentes.

Acima de tudo, diversos personagens-chave têm fortes objeções a alguns aspectos da implementação atual de Nigel. APIs mudaram e outras partes do código, que haviam sido criticadas no passado, aparentemente permaneceram intocadas nos patches de Nigel. A situação como um todo é uma verdadeira bagunça; mas apesar das discordâncias, e apesar de haver mais trabalho sendo jogado sobre os ombros de Nigel do que talvez fosse necessário, as discussões são bastante civilizadas e todos parecem mais ou menos dedicados à obtenção do mesmo resultado.

Suspender para o disco rígido um sistema em operação é um problema inerentemente desajeitado e desagradável, pois não há um modo seguro para confirmar que todo o hardware estará no mesmo estado após a restauração. Por essa razão, qualquer tentativa de suspender um sistema ativo estará coalhada de palpites, estimativas e feitiçaria. É maravilhoso que Nigel, Pavel e os desenvolvedores do Linux em geral vejam a importância de atacar esses problemas que talvez nunca tenham solução definitiva, apenas uma esperança de aprimoramento de versão para versão. ■

OSDL Documentation Set

Alguns interessantes projetos de documentação vieram à luz. Timothy D. Witham, a pedido de Andrew Morton e Alan Cox, estabeleceu um repositório para documentação digital e impressa obtida de distribuidores. Diferente de projetos como o Linux Documentation Project, o OSDL Documentation Set trata principalmente de arquivar documentos obscuros, incluindo manuais e especificações de hardware usados para desenvolver o Linux. Embora a documentação em papel reunida até agora pelo OSDL não seja muita, a documentação digital está se tornando bastante extensa, cobrindo uma ampla gama de referências, de ACPI a vídeo. ■

Documentação do *ioctls*

Nesse meio tempo, Edward Falk, do Google, arriscou-se na tarefa extremamente aterrorizante de documentar os *ioctls* (*Input and Output Controls* – Controles de Entrada e Saída) do Linux. Isso é algo que muitas pessoas se propuseram a fazer ao longo dos anos, mas onde quase não se pôde obter resultados identificáveis. Edward, porém, parece estar fazendo progressos. O problema fundamental em qualquer projeto que vise a documentação dos controles de entrada e saída é que qualquer desenvolvedor de drivers pode criar sua própria chamada aos *ioctls*, com um conjunto totalmente único de entradas de dados e comportamentos

Até mesmo catalogar todos eles é um pesadelo em potencial, considerando-se que nem todos os drivers do Linux são inclusos nos fontes oficiais do kernel. Há *ioctls* espalhados pelos quatro cantos do globo. Ir atrás de todos eles seria algo insano. Ed decidiu atacar essa enorme quantidade de implementações de *ioctls* em pequenos grupos específicos.

Começou com os *ioctls* do IDE e passou para os *ioctls* de CD-ROM, submetendo patches de documentação que eram aplicados com sucesso. Talvez seu trabalho inspire outras pessoas a adotar outras áreas do kernel usando os *ioctls*. Um fator de alívio no imensurável tamanho do projeto é que a documentação oficial pode se restringir a documentar apenas os *ioctls* realmente definidos no kernel oficial. Mesmo assim, esta ainda é uma tarefa hercúlea, mas não tão impossível quanto se pensava.

Outro fator de abrandamento é que a criação de novos *ioctls*, ao menos dentro do código fonte oficial do kernel, é fortemente desencorajada. Enquanto personagens como Alan Cox previnem contra o ódio cego dirigido aos *ioctls*, a maioria dos grandes desenvolvedores ainda os odeia e deseja que morram secos, pra não dizer algo pior. Realmente, após os tremendos avanços feitos em projetos como o UDEV (diretório `/dev` dinâmico) e SysFS, a necessidade de coisas como *ioctls*, *ProcFS* e outras interfaces legadas entre o núcleo do sistema operacional e o espaço do usuário realmente diminuiu muito. ■

Mantenedores

O problema da manutenção está constantemente passando por refinamentos e outras mudanças orgânicas. Disputas são resolvidas, precedentes são abertos, políticas são criadas, recriadas, jogadas no lixo e criadas novamente. Desenvolvedores vêm e vão, a opinião popular diverge e vira e mexe algo bacana acontece.

Um problema que surgiu recentemente é se listas de mensagens moderadas devem ser incluídas no arquivo `MAINTAINERS`. Essa questão nunca teria surgido se a lista de mensagens *linux-kernel* fosse moderada; mas Linus Torvalds sempre acreditou que qualquer pessoa deveria ser capaz de postar seus relatórios de falhas e questões para a lista, de modo que os desenvolvedores do kernel tivessem acesso à informação de que necessitassem para resolver falhas e melhorar o código no geral. Por outro lado, os administradores da lista recentemente decidiram que qualquer pessoa que proponha constantemente questões *off-topic* ou alimente *flame wars* pode ser proibida de postar na lista. Ops, a situação não é assim tão simples...

No passado, a listagem de listas de discussão moderadas no arquivo `MAINTAINERS` foi desprezada, com a justificativa de que qualquer um deveria ser capaz de postar seus relatórios de falhas nas listas de discussão de sub-projetos específicos do kernel, assim como se pode fazer na lista principal *linux-kernel*. E, recentemente, Domen Puncer tentou enviar um patch que removeria algumas listas moderadas desse arquivo.

Porém, Alan Cox e outros evitaram isso, com a justificativa de que, mesmo se um não-membro pudesse postar numa lista, ele ainda gostaria de ser associado dela e postar novamente. Embora não seja o grande assunto do momento para os desenvolvedores do kernel, o problema está longe de ser resolvido e provavelmente passará ainda por diversos refinamentos antes de assentar.

Num movimento menos controverso, Jeff Garzik recentemente começou a marcar certos projetos como desaprovados nos scripts de configuração. Sua idéia é cuidar para que nenhuma pessoa interessada nesses projetos tenha a chance de protestar contra a remoção deles quando chegar a hora. Projetos que estão quebrados, abandonados ou que realizam a mesma função que outros são atualmente alvos para a eliminação.

A manutenção do MCA (*Micro Channel Architecture*) mudou de mãos recentemente. Adrian Bunk enviou algumas limpezas para esse código, e David Weinehall, o mantenedor anterior, disse que passou o bastão para James Bottomley, que deve receber todos os patches para MCA a partir de agora. James, por sua vez, postou um patch para o arquivo `MAINTAINERS`, tendo se listado como mantenedor oficial do MCA. Ele admitiu que não estava exatamente nadando em hardware MCA e, assim que declarou isso, Alan Cox enviou a ele uma enorme pilha de placas, computadores e componentes que usam a tecnologia. Parece que agora James terá bastante intimidade com o código MCA.

Num contexto completamente diferente, Adrian tentou remover Michael H. Warfield da manutenção da órfã placa Computone Intelliport Multiport. Andrew Morton não gostou da idéia, e Jim Nelson, em particular, achou que a inclusão do antigo mantenedor no arquivo `MAINTAINERS` permitiria, ao menos, que novos mantenedores em potencial percebessem de quem deveriam receber as rédeas. Mas Adrian argumenta fortemente que a intenção do arquivo `MAINTAINERS` é que os desenvolvedores o consultem para descobrir para onde devem mandar seus patches.

A idéia de Adrian é que, caso necessário, um arquivo alternativo com a listagem do status atual de todos os projetos também poderia ser mantido. Mas isso exigiria muito esforço. Não está clara a forma como ele seria manejado, mas com certeza ouviremos falar de mais nomes antes que esse problema seja resolvido. ■

Sobre o autor

*A lista de discussão **linux-kernel** é o centro do desenvolvimento do kernel Linux. O volume de tráfego é imenso e se manter em dia com todo o processo é uma tarefa humanamente impossível. Uma das poucas pessoas corajosas o suficiente para aceitá-la é Zack Brown, que já publica um "resumão semanal" das discussões, na forma da lista **kernel-traffic**.*

Esta coluna mensal manterá você informado sobre as últimas novidades e decisões relativas ao kernel, selecionadas direto da fonte e resumidas pelo próprio Zack.