Notícias do Kernel

Reunificação do Software Suspend

Em setembro de 2003, Patrick Mochel e Pavel Machek tiveram uma discussão acalorada sobre o trabalho sendo feito no *Software Suspend*, um mecanismo que possibilita salvar o estado de um sistema em execução e retornar a ele depois de uma reinicialização. Durante a discussão, Patrick chegou a ponto criar um *fork* do código de Pavel e iniciar seu próprio projeto, o *pmdisk*.

O ato de criar um fork de um projeto de código aberto, apesar de perfeitamente legal, traz fortes elementos da cultura ao redor. Existem protocolos sociais que dizem como ou quando um projeto deve ser dividido. Uma razão legítima é quando o mantenedor é irresponsável, ou, sem motivo, rejeita as contribuições de um desenvolvedor. Patrick acreditava que Pavel estava rejeitando correções razoáveis, e tornando impossível contribuir com o swsusp.

Os projetos continuaram desenvolvendo-se paralelamente e anunciando seus progressos na lista de discussão *linux-kernel*, o que alimentou a discórdia; gradualmente, Patrick começou a diminuir o ritmo de desenvolvimento. Finalmente, em julho desse ano, depois de algumas conversas particulares com Pavel, eles decidiram mesclar seus trabalhos, com Pavel como o líder do novo projeto. Ao divulgar a reunificação, Patrick pediu desculpas pelo fork e fez as pazes com Pavel. Os dois vêm trabalhando harmoniosamente desde então.

Essa saga se junta ao ranking dos forks que se reintegraram. O que é algo relativamente raro; a maioria dos forks ou tomam o lugar do projeto original (como no caso do X.org, que praticamente varreu o XFree86 do mercado), ou continuam indefinidamente como um projeto concorrente. Raramente ambos os projetos contêm contribuições valiosas que depois são reunidas. Exemplos de tais reunificações incluem projetos importantes como o compilador GNU C (gcc) e a biblioteca Glibc. ■

Cryptoloop por um fio

Desde o kernel 2.6.3, Andrew Morton vem considerando remover o cryptoloop do kernel 2.6. O Cryptoloop veio à tona em julho de 2003 a partir do trabalho de Andries Brouwer, durante a série de desenvolvimento do kernel 2.5, para permitir que sistemas de arquivos criptografados sejam montados sobre loopback. Já na época Andrew não estava muito de acordo, e em agosto deste ano James Morris foi bem recebido ao apresentar um patch que implica na remoção completa do cryptoloop.

Um problema em remover o cryptoloop, ou qualquer recurso do kernel 2.6, é que atualmente o kernel está no ciclo estável, e até mesmo uma pequena mudança visível para o usuário está fadada a encontrar resistência severa. Mas a visão de Andrew para a série estável traz algumas nuances que já existiam, de uma maneira informal, durante séries estáveis anteriores. Entre elas está a decisão de permitir às distribuições serem as principais fontes de kernels estáveis. O kernel principal, mesmo durante a série estável, irá em direção à abundância de recursos e um alto nível de eficiência. A estabilidade não será abandonada, mas não será o único objetivo, como foi no passado.

Andrew falou que seu novo método irá adaptar-se à medida em que o desenvolvimento do kernel continuar, e à medida em que os desenvolvedores do kernel e usuários expressarem suas opiniões sobre o assunto. O objetivo, aparentemente, é aperfeiçoar o processo de desenvolvimento ao longo do tempo, inclusive a tradicional divisão entre a série estável e de desenvolvimento.

A dependência nos kernels produzidos pelas distribuições é uma nova idéia no desenvolvimento do kernel do Linux. Historicamente, desenvolvedores tendem a acreditar que o código-fonte oficial deveria ser o suficiente para o uso em qualquer sistema. Sem dúvida, muitas versões do kernel continuarão a ser usadas desta forma, mas a relação entre os desenvolvedores do kernel e desenvolvedores das distribuições parece estar passando por uma mudança na direção de uma descentralização maior.

Dentro desse contexto, o cryptoloop é um recurso que, com seus bugs, problemas de segurança e falta de um mantenedor, será retirado do kernel com menos cerimônia do que o de costume. Mas, em respeito às tradicionais preocupações quanto à solidez e confiabilidade da série estável, Andrew parece estar inclinando a removê-lo na série de desenvolvimento 2.7. ■

Reiser4 entrando para o 2.6

O sistema de arquivos Reiser4 foi aceito no kernel 2.6.8.1-mm2. O criador deste sistema de arquivos, Hans Reiser, há muito tempo faz pressão para a sua inclusão no kernel oficial. Sua entrada na série -mm, mantida por Andrew Morton, o coloca a apenas um passo de atingir este objetivo.

A comunidade de desenvolvedores e usuários do Reiser está empolgada quanto a esta versão, já que o Reiser4 representa o resultado de muito trabalho, e um significante distanciamento da versão 3 do ReiserFS.

SOBRE O AUTOR

A lista de discussão *linux-kernel* é o núcleo das atividades de desenvolvimento do kernel Linux. O volume de tráfego é imenso e manter-se em dia com todo o processo é uma tarefa praticamente impossível para uma só pessoa. Uma das poucas almas corajosas o suficiente para aceitá-la é Zack Brown.

Esta coluna mensal manterá você informado sobre as últimas discussões e decisões, selecionadas e resumidas pelo próprio Zack, que já publica um resumo semanal há vários anos sob a forma da lista *Kernel Traffic*. A LM agora lhe dá acesso à essência das atividades de desenvolvimento do kernel, direto da fonte.

O Reiser4 não é diretamente compatível com seu predecessor. Antes de atualizar seu sistema de arquivos você deve fazer um backup dos seus arquivos importantes, criar um novo sistema de arquivos na partição, o que implica na perda de todos os dados nela armazenados, e depois restaurar o backup. Apenas os recursos básicos foram incluídos nesta versão; recursos mais exóticos como acesso a múltiplos pequenos arquivos com uma única chamada de sistema foram postergadas até que o código central seja aceito.

Como o venerável sistema de arquivos ext2 continua sendo o mais comumente utilizado, será interessante observar os progressos no desenvolvimento do ReiserFS, Ext3, XFS e outros, já que cada um tenta colher os frutos de suas respectivas visões, e alcançar os usuários que mais podem se beneficiar delas. ■

Problemas no *Scheduler*

Um grupo de desenvolvedores tem ficado cada vez mais frustrado com o estado do agendador de processos (*scheduler*) no Kernel 2.6. Ingo Molnar e Arjan van de Ven encontraram dificuldades em fazer trabalhos de áudio complexos devido aos atrasos (*lag*) e pulos causados pelo agendador. A sede por desempenho em tempo real do Linux existe entre os desenvolvedores há mais de uma década, e mesmo que muito trabalho tenha sido feito desde aquela época, há ainda muito trabalho a ser feito. Ingo e Arjan conseguiram criar um patch que adiciona muitos *scheduling points* ao longo do kernel, desta forma reduzindo a quantidade de tempo entre o reagendamento de processos.

O problema com esse método é que ele tende a desconsiderar o trabalho que já foi, e que continua a ser feito, no scheduler. Mais especificamente, Robert Love e Andrew Mortom acreditam que as mais prováveis causas dos problemas do agendador são erros consertáveis, que não necessitam de uma nova infraestrutura.

A divergência é significante, independente dos resultados. O fato de que dois sujeitos poderosos no desenvolvimento do kernel não conseguem chegar a um acordo ilustra a notória dificuldade no desenvolvimento do scheduler do Linux. De fato, nos anos recentes, surgiu uma verdadeira leva de diferentes implementações do scheduler, à medida que os desenvolvedores exploram várias teorias. Foi até sugerido que o scheduler a ser usado deveria ser especificado durante a compilação, permitindo aos usuários escolherem dentre qualquer número de implementações alternativas.

Até agora essa idéia não teve muita adesão entre os mantenedores do kernel (Linus, Andrew, etc.), provavelmente porque o agendador é uma peça tão fundamental no sistema. Oferecer várias implementações que competem entre si em um kernel oficial poderia dificultar a tarefa de confirmar que uma determinada série do kernel caminha em direção à estabilidade. Ainda assim, é fascinante observar o esforço deles para melhorar suas soluções para este problema tão complexo. ■

Espaço do Kernel e Espaço do Usuário

Surgiu um novo mecanismo para comunicação entre o espaço do usuário e o espaço do kernel, e os desenvolvedores estão agora debatendo como mantê-lo sob controle afim de que não se torne outro ProcFS ou I/O Control (*ioctl*), com um comportamento inteiramente descontrolado e trancado dentro do kernel devido à necessidade de compatibilidade retroativa.

Robert Love, Arjan van de Ven e Kay Sievers implementaram um sistema assíncrono de notificação, ou uma camada de "eventos do kernel", que envolve o link de rede para permitir que os eventos do kernel sejam transmitidos ao espaço do usuário. A prova de conceito inicial implementada pelo trio focou-se na detecção e notificação da temperatura do processador. Teoricamente, quaisquer números de tais eventos podem ser identificados e notificados ao utilizar esse mecanismo.

O problema é como evitar que a camada de eventos se sobreponha ao SysFS, ou como integrá-la completamente ao SysFS, se é isso o que precisa ser feito. No momento, ninguém tem muita certeza quanto à organização apropriada, embora claramente o recurso seja poderoso e potencialmente bastante útil. Mas se ele se tornar outro diretório /proc, com convenções não uniformes aglomerando-se arbitrariamente, daqui a cinco anos – ou menos – os desenvolvedores podem desejar nunca terem posto os olhos nesse código. Felizmente as lições aprendidas com /proc, ioctl(), e /dev aparentemente foram bem assimiladas, e a discussão sobre como evitar este pesadelo futuro já começou. ■

GPL na Justiça

Por várias vezes os desenvolvedores especularam se a GPL – *GNU General Public License* – seria capaz de se sustentar na justiça, mas até agora nenhum caso veio à tona. A versão 2.0 da GPL é a licença utilizada pelo kernel Linux e por milhares de outros projetos de Software Livre. Se ela falhar no teste nos tribunais, as conseqüências podem ser desastrosas para o mundo do código aberto, e para praticamente qualquer um que utilize ferramentas baseadas em Software Livre.

No mês de abril, um tribunal em Munique concedeu um mandato de injunção preliminar contra a Sitecom, em benefício do projeto de código aberto Netfilter. O projeto acusou o produtor de roteadores de utilizar seu código no firmware de seus produtos, mas redistribuí-lo apenas em forma binária, o que vai contra a licença GPL adotada pelo projeto. Sendo apenas uma injunção preliminar, as ações do tribunal não estabelecem um precedente. Elas apenas indicam que o tribunal acredita que as reivindicações do projeto Netfilter serão justificadas; nenhum outro caso pode usar esta injunção para defender suas próprias reivindicações quanto à violações da GPL.

Todos os olhos estarão nesse caso, embora seja possível que a Sitecom evite maiores problemas legais adequando-se às condições da GPL. A Sitecom está se protegendo ao afirmar que não ela, mas sua empresa controladora, seria o alvo apropriado de qualquer processo judicial potencial. Se esse for o caso, os procedimentos podem arrastar-se por um bom tempo. Ao que tudo indica, a Sitecom irá simplesmente publicar o código-fonte ou retirar seu produto do mercado, deixando a GPL ainda sem ser testada. ■