Crônicas do kernel

Zack Brown

Problemas dentro do kernel e infinitas possibilidades fora dele. Quem disse que amadurecer é fácil?
por Zack Brown

Big Kernel Lock

Linus Torvalds postou um *patch* para desfazer uma grande alteração no sistema de bloqueios (*locks*) do kernel. O *grande bloqueio do kernel* (BKL, para os íntimos) havia sido alterado a partir de um *spinlock*, com alguns problemas de latência, para um semáforo. Isso representou um grande avanço para quem desenvolvia monitores de saúde para hospitais (e também para quem aprecia jogos de tiro em primeira pessoa).

A solução do semáforo também ocasionou grandes problemas de desempenho em certos *benchmarks*, e as soluções para eles não resolveram tudo. Então, Linus decidiu que a forma simples seria voltar à forma antiga.

Linus também indicou que a única forma de contornar o problema do spinlock seria eliminar inteiramente o BKL. Ingo Molnar, auto-declarado "viciado em latência", não gostou disso, mas aceitou a colocação de que a solução seria cortar o BKL pela raiz. Acontece que eliminar o BKL é difícil! Foram necessários os talentos de Alan Cox (palavras de Ingo) para se aventurar no labirinto semântico e entender o que e como mudar nessa área. Segundo os cálculos de Ingo, até mesmo com desenvolvedores como Alan já mergulhados nessa tarefa, o ritmo atual exigiria mais de dez anos para remover o BKL. Ingo explicou o problema, dizendo que "suas dependências são totalmente desconhecidas e invisíveis e tudo está perdido nos últimos 15 anos de alterações no código. Tudo isso acabou criando uma espécie de medo, incerteza e dúvida relativas ao BKL: ninguém o conhece direito, ninguém se atreve a tocá-lo e o código pode quebrar silenciosa e sutilmente se o BKL estiver errado".

A solução de Ingo foi criar uma árvore *git* especificamente para receber as particularidades mais assustadoras do BKL, com o objetivo final de facilitar a remoção em massa do código. Um dos primeiros passos principais foi extraí-lo do código central do kernel e mover toda a sua feiúra para algum lugar em que possa ser completamente substituído (ao menos teoricamente) no futuro, mudando o comportamento ao longo de todo o kernel assim que se criar uma implementação melhor. Como disse Ingo, "uma vez que essa árvore se estabilize, a eliminação do BKL pode ser feita da forma normal de se eliminar grandes bloqueios – empurrando-os para os subsistemas, substituindo-os por bloqueios específicos de cada subsistema, dividindo-os e, finalmente, eliminando-os. Já fizemos isso várias vezes no passado e há muitos desenvolvedores capacitados que pode atacar tais problemas".

Andi Kleen gostou do plano, mas preferiu não esperar. Em vez de ter uma árvore git separada para todas as alterações, por que não efetuá-las no ramo oficial? Linus e vários outros hackers também gostaram de ver Ingo trabalhando nisso e ofereceram inúmeras sugestões.

Porte do Linux para... tudo?

Octavian Purdila, Stefania Costache e Lucian Adrian Grijincu estão essencialmente tentando portar o Linux para rodar em qualquer outro projeto de programação. Eles o chamam de *Linux Kernel Library Project*, e o objetivo é converter vários códigos, como o *Virtual File System* do Linux, por exemplo, em bibliotecas genéricas que permitam que qualquer pessoa as insira em seus projetos.

Caso os programadores tenham êxito, qualquer sistema operacional será nativamente capaz de suportar qualquer recurso do Linux, bastando usar essa biblioteca. A abordagem é criar a biblioteca como um porte direto do Linux para uma arquitetura virtual que depois poderá ser usada para qualquer objetivo.

Octavian, Stefania e Lucian já obtiveram algum progresso e estão procurando voluntários para ajudá-los a manter o porte atualizado em relação ao kernel. ■

Sobre o autor

A lista de discussão *Linux-kernel* é o núcleo das atividades de desenvolvimento do kernel. **Zack Brown** consegue se perder nesse oceano de mensagens e extrair significado! Sua *newsletter Kernel Traffic* esteve em atividade de 1999 a 2005.

NovaForge™

Nós conectamos nossos Clientes a nossos Centros de Competências de Software Livre

NovaForge, no centro da abordagem Industrial para Desenvolvimento de Sistemas da Bull.

O NovaForge é um poderoso conjunto de ferramentas e serviços amplamente testados e projetados para reduzir o esforço, otimizar custos de gestão e cronogramas, garantindo a qualidade dos produtos finais em Projetos de Desenvolvimento de Sistemas. O NovaForge foi concebido para ser utilizado em Projetos de Desenvolvimento e Atualização de Aplicações em ambientes J2EE, PHP e .net , na manutenção de aplicações desenvolvidas por terceiros e para o teste profissional e integrado dos sistemas.

BULL

Architect of an Open World™

www.bull.com