Notícias do kernel

A novela do BitKeeper...

...terminou, afinal. Linus Torvalds parou de usar o sistema de controle de versões *BitKeeper*. Como decorrência disso, temos observado o esforço da comunidade de desenvolvedores para encontrar um substituto à altura da função – esforço esse fascinante de se observar. A decisão foi repentina e tanto Linus como os outros desenvolvedores estão ocupados vasculhando os quatro cantos em busca de uma maneira de continuar o desenvolvimento do kernel da maneira como estavam acostumados – ou seja, com um volume de código estonteante. E é precisamente esse alto volume de trabalho que Larry McVoy, CEO da BitMover, disse ser impossível de ser acomodado por qualquer das ferramentas de controle de versão desenvolvidas pela comunidade, pelo menos a médio prazo. Esperemos para ver se o homem que fabrica o BitKeeper tem razão ou se é apenas papo. Nesse caso, provavelmente um software desenvolvido pela comunidade tomará seu lugar.

Obviamente, é bastante provável que Linus e seus comparsas encontrem (ou mesmo desenvolvam) algo que funcione para eles, da mesma forma como o venerável sistema de envio de *patches* funcionou nos velhos tempos – atendia ao Linux, mas não podia ser aplicado como solução de uso geral para outras comunidades de desenvolvedores. Com isso, uma das equipes com maior volume de trabalho, hoje, é justamente a que desenvolve o conjunto de scripts 'temporários' para que o kernel possa continuar sendo desenvolvido enquanto Linus procura por uma solução permanente. O pai de todos já avisou: está pensando até mesmo em adotar o *Subversion* como solução paliativa caso algo concreto não seja encontrado nos próximos dias, embora tanto ele como os desenvolvedores do Subversion concordem que essa não é a melhor ferramenta para uso com o kernel Linux. Se isso acontecer, é porque a coisa degringolou mesmo – mas é bastante possível e nem tão improvável assim.

De qualquer forma, um número razoável de aplicativos promissores estão sendo preparados para preencher a lacuna. Embora o *arch* não seja consenso, em grande parte devido a seu intratável mantenedor, há um grande número de projetos que são ou desdobramentos (*forks*) diretos do *arch* ou reimplementações de seus recursos básicos. Um deles é o *bazaar-ng*, de autoria de Martin Pool, um "mano sangue bom" no desenvolvimento do kernel. Ele vem trabalhando como um alucinado para tornar seu rebento suficientemente usável para os desenvolvedores do kernel – pelo menos como solução 'interina'.

Outro forte candidato é o *monotone*, um sistema de controle de versões distribuído que Linus já tinha elogiado antes, classificando-o de "a melhor das alternativas livres". Mas embora o monotone seja um sistema indubitavelmente ótimo, ainda precisa de bastante suor para ser minimamente útil para as necessidades dos desenvolvedores do kernel. Não ficou claro, ainda, se a maciça atenção que o projeto recebeu recentemente está ajudando ou atrapalhando o esforço frenético de sua equipe de desenvolvimento.

Uma estranha – e fascinante – abordagem do problema está no sistema de arquivos *git*, desenvolvido por... Linus Torvalds! Ele já está funcionando em termos aceitáveis, mas isso não deve ser interpretado como "já está pronto pro povo lá fora". Também não deve ser inferido que Linus quer criar um sistema de controle de versões no sentido tradicional da expressão. O que quer que esteja saindo da mente alucinada de Torvalds, será algo novo. Ele sugeriu, entretanto, que um sistema tradicional de controle de versões poderia ser construído usando o git como alicerce – mas ele próprio não moverá uma palha nessa direção; apenas implementará os recursos que julgar importantes. Ao mesmo tempo, Linus espera largar o fardo do desenvolvimento do git sobre costas alheias, mantendo as rédeas de mantenedor e reservando-se o direito de aceitar (ou recusar) patches. ■

CPUs Geode finalmente no kernel

Kianusch Sayah Karadji tem mantido há bastante tempo um patch que dá ao kernel 2.6 suporte às CPUs Geode. O patch é essencialmente independente e estava, até então, disponível apenas em seu site pessoal. Mas como cada vez mais usuários escreviam pedindo que fosse incorporado ao kernel oficial, Kianush finalmente decidiu enviá-lo para inclusão na árvore principal. O patch deu entrada no final de fevereiro – tarde demais para entrar na versão 2.6.11, lançada no dia 2 de março – mas só apareceu na 2.6.12-rc1 algumas semanas depois. Se nada

de anormal for identificado no código, vai aparecer definitivamente na versão estável 2.6.12 assim que for lançada. Kianusch tem ajudado os desenvolvedores do kernel com scripts desde 2002, mas essa é sua primeira contribuição ao kernel propriamente dito. ■

❒ Hotplug Redux

Greg Kroah-Hartman tirou da cachola (e do papel) um subsistema de *hotplug* que roda no espaço do usuário. Chamado de *hotplug-ng*, a intenção é substituir o sistema atual, baseado em scripts do *Bash*, por um binário compilado de tamanho diminuto. Greg reimplementou a coisa em C para evitar a necessidade de se ter o interpretador Bash instalado, bem como para reduzir as necessidades de espaço em disco e memória de todo o sistema. A velocidade conseguida, entretanto, é de longe o maior benefício alcançado pela reimplementação. Em testes empíricos e sem qualquer medição precisa, os scripts em Bash levaram de dois a três segundos para fazer os dispositivos funcionarem, enquanto o binário compilado levou menos de um segundo. Nem o próprio Greg tinha notado a relevância de seu feito e só percebeu que era algo realmente formidável quando o pessoal dos sistemas embarcados começou a cantar e dançar de alegria. Os desenvolvedores do SUSE LINUX, por exemplo, detectaram esperas de até 40 segundos que podem ser reduzidas para dois ou três se a reimplementação de Greg for usada. ■

❒ Cartões de memória SD

Pierre Ossman mostrou a seus pares seus esforços para tornar os cartões de memória do tipo *Secure Digital* (SD) compatíveis com Linux. Esses cartões podem armazenar vários gigabytes e se propõem a facilitar o transporte de arquivos multimídia entre uma grande variedade de tipos de hardware. Esses cartões também possuem um sistema de controle de direitos autorais e propriedade intelectual: há restrições para os tipos de dados que podem ser extraídos do cartão. Alguns desenvolvedores do kernel se mostraram empolgados por ver algo sendo produzido nesse terreno, mas infelizmente Pierre não teve acesso a informações técnicas suficientes para terminar o trabalho e, em especial, para ter seu código incluído no kernel oficial. Isso se deve em parte a questões jurídicas: será que há permissão legal para escrever drivers para os cartões SD? ■

❒ Novo subsistema central timeofday

John Stultz vem trabalhando num subsistema de gerenciamento de hora no sistema que, para começar, é independente de plataforma. Com isso ele espera reduzir a cruel complexidade e a infame duplicidade (ou, antes dizer, multiplicidade) de código que torna essa área do kernel uma verdadeira esfinge para os desenvolvedores. Mas tomar uma parte do código que aparentemente depende da plataforma de hardware e torná-la genérica o bastante para funcionar em qualquer lugar é um desafio digno dos personagens de Júlio Verne. Algumas das arquiteturas têm porções desse código no espaço do usuário, enquanto outras implementam tudo no kernel. Além disso, qualquer código que John venha a apresentar deve ter um desempenho igual ou superior ao código que ele pretende substituir. Para aplacar o sofrimento que toda mudança acarreta, John criou um "gancho" opcional para o novo subsistema – cada arquitetura fará uso desse *hook* quando mais lhe aprouver. ■

❒ SysFS mostra os primeiros sinais de fadiga

Uma tentativa recente de corrigir a localização de um diretório mal-posicionado no *SysFS* deu com os burros n'água: um número não desprezível de outros subsistemas já esperam que o diretório exista naquela posição. Esse é, precisamente, o tipo de situação que fez do sistema de arquivos `/proc` o pesadelo dos desenvolvedores e usuários. O conjunto de mantenedores do SysFS tentou ao máximo evitar as armadilhas (e o destino) de seu antecessor *ProcFS*, mas esse incidente inexoravelmente indica que os mesmos problemas ocorrerão no futuro.

Os desenvolvedores do SysFS estão a par do histórico desastroso do ProcFS, dos *ioctls* e do diretório `/dev`: todos os três viraram terríveis "chupa-cabras". Em vista deste passado horripilante, é de se esperar que o SysFS tenha vida muito mais longa que seus ancestrais. Com um pouco de otimismo podemos dizer que, se o "capenga" ProcFS pôde ser usado por décadas, talvez o SysFS, mais robusto, esteja em plena vida útil daqui a alguns séculos.

O grande problema encarado por todos esses sistemas (incluindo o SysFS) é a maneira como uma interface consistente (ou seja, que faça um mínimo de sentido) deve ser apresentada ao usuário. O paradoxo é que essa interface tem de se manter simples e, ao mesmo tempo, ser compatível com um número cada vez maior de serviços, número esse que cresce exponencialmente. A quantidade de hardware a ser suportado cresce sem limites – e todos eles precisam ser representados no `/dev` de uma forma que os usuários comuns possam reconhecer e usar. Os drivers e outros pontos do kernel crescem, mudam, são divididos em categorias diferentes (e, por vezes, dividem uma mesma categoria com drivers que, no passado, estavam em outras) e, ainda assim, devem aparecer de uma forma consistente, útil e à qual o usuário já esteja acostumado. O problema existe desde tempos imemoriais, quando o Unix ainda não era escrito na linguagem C, e continuará a existir até que a última variante de Unix seja extinta. ■