Uma olhada atualizada no Software Livre e seus asseclas

Projetos na incubadora

O conhecido programa de configuração do SuSE, o YaST2, ajudará usuários do Debian no futuro.E se você está procurando um modo de relaxar ao término de suas incumbências de administrador de sistema, dê uma olhada em "Battle for Wesnoth."

por Martin Loschwitz

É raro que a seção *Projetos na Incubadora* fale sobre algum jogo. A razão para isso é a falta de jogos novos e ao mesmo tempo interessantes para Linux. Para expiar essa falta de atenção dada aos jogos, este mês vamos nos concentrar num jogo de estratégia chamado *The Battle for Wesnoth* (**figura 1**). O jogo é leve o bastante para rodar em hardware mais antigo e oferece um projeto interessante que compensa muito bem a falta de gráficos 3D sofisticados.

Falarei também dos esforços do Debian para incorporar o YaST2, originalmente desenvolvido para o SUSE LINUX, e descreverei o sistema de arquivos SSHFS. Terminamos com uma novidade sobre o Debian Sarge.

Estratégia de jogo

A rainha de Wesnoth está tramando contra seu marido com o objetivo de colocar um amiguinho no trono. Para isso, ela selou pactos secretos com os membros da corte real. O mago real, que é leal ao rei, descobre o complô e se volta contra a rainha para salvar o herdeiro legítimo do trono. Se você se lembra de ter jogado *Warlords* nos bons tempos do Amiga, certamente vai se sentir em casa em The Battle for Wesnoth.

Há dois modos típicos para jogos de estratégia. O modo *realtime* (tempo real) significa que o jogo continua mesmo se não houver jogadores, enquanto o modo *round-based* (em turnos) espera que os jogadores façam suas jogadas. O Wesnoth usa esse segundo modo. Diferente de muitos outros jogos de estratégia, o jogo inicia com uma coleção completa de edifícios e, portanto, não se espera que os jogadores se envolvam em projetos de construção.

Colírio para os olhos

No mapa que mostra o reino de Wesnoth fica óbvio que os desenvolvedores colocaram bastante amor e carinho na criação do som e dos gráficos. Assim, o Wesnoth escapa do destino de muitos jogos em que um enredo ou jogabilidade excelentes sofrem com gráficos mal-desenhados. Bons gráficos não fazem um bom jogo sozinhos, mas ajudam.

Uma grande variedade de personagens e grupos, incluindo arqueiros e cavalaria com diversas características, contribuem para interessantes batalhas. Algumas unidades lutam melhor à noite, outras durante o dia. Os soldados têm diferentes níveis de energia vital, custos de manutenção e mesmo de inteligência. A inteligência é expressa pela velocidade com a qual as tropas aprendem com sua experiência na batalha. Xamãs cuidam dos feridos.

O jogo diferencia combate de curta e de longa distância. Se estiver na sua vez, você também deve escolher o tipo

Figura 1: Na Batalha por Wesnoth, os jogadores têm que proteger a corte e o país de uma rainha do mal.

de armas que usará no ataque. A defesa é automática, desde que a unidade que esteja sofrendo o ataque tenha os recursos necessários. Se a unidade não possuir uma arma para combate corpo-a-corpo, pode estar condenada. Como em muitos outros jogos, o objetivo é apossar-se de tantas edificações quanto possível e derrotar o inimigo!

Um tutorial ajuda os jogadores a entrar no cenário de pesadelo do combate medieval e demonstra as etapas mais complicadas. The Battle for Wesnoth é frugal em suas exigências de hardware e software; necessita apenas da biblioteca SDL. E se você não gostar dos mapas que vêm com o jogo, pode ligar o editor de mapas e desenhar o seu próprio. Interessou? É só olhar a homepage [1] para maiores informações. ■

YaST 2 sem SUSE

O YaST 2 é provavelmente o programa de configuração que, sozinho, dividiu a comunidade Linux mais do que qualquer outro ao longo dos últimos anos (**figura 2**). Essa invenção da SUSE foi um espinho no pé da comunidade do FOSS (*Free and Open Source Software* – Software Livre e de Código Aberto), já que não era desenvolvido sob uma licença livre e, por isso, contradiz a filosofia básica subjacente ao Linux. Ao mesmo tempo, o YaST 2 oferece ao usuário uma interface gráfica intuitiva, que é apreciada por especialistas e recém-chegados.

Os que ainda não usaram o SUSE LINUX até agora, não importa a razão, tiveram de se virar sem o YaST 2. Sem uma licença livre, não havia como portar o YaST 2 para outras distribuições. Felizmente, essa situação mudou. Em seguida à aquisição da SUSE, a Novell anunciou em março de 2004 que lançaria o YaST 2 sob uma licença livre.

A explicação dada pela Novell faz sentido: o YaST 2 da Novell oferecerá uma interface de configuração uniforme para todos os sistemas Linux. Nos meses que se seguiram à remoção das restrições de licenciamento, não houve qualquer reação óbvia. Mas em novembro de 2004 Mario Fux começou a pensar alto sobre portar o YaST 2 na lista de discussão Debian-desktop. Afinal de contas, o Debian é carente em software de configuração. Enquanto o Mandrake e o Fedora têm diversas ferramentas com interface gráfica para incumbências administrativas críticas, os usuários do Debian precisam recorrer ao console. O Debian carece até de ferramentas em modo texto para algumas tarefas, o que força os usuários a abrir o editor de textos e modificar diretamente os arquivos de configuração, em alguns casos.

Mario Fux surpreendeu a audiência com um anúncio interessante. Disse que gostaria de dar uma recompensa e doar hardware para o desenvolvedor que portasse o YaST2 para o Debian. Fux deixa sua motivação bastante clara; pretende fazer do Debian um sistema operacional para usuários desktop, aconteça o que acontecer. Os desenvolvedores do Debian estão desde então discutindo os prós e os contras de se portar o YaST. Gerenciamento de software é visto como

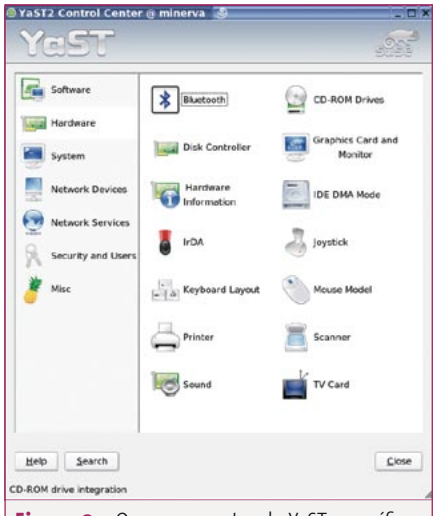

Figura 2: Os componentes do YaST específicos ao SuSE têm de ser removidos para permitir que o YaST seja portado para outras distribuições.

um assunto delicado, já que o SUSE e, portanto, o YaST 2, são baseados em pacotes RPM.

Um pequeno grupo de desenvolvedores formou-se para apoiar Mario Fux; atualmente esse grupo se ocupa de portar o YaST. Uma sessão de *chat* no início de fevereiro revelou as últimas notícias. O planejamento está num estágio bastante avançado e é provável que o projeto esteja seriamente encaminhado num futuro próximo. Mario Fux contatou o principal desenvolvedor do YaST 2 no SUSE, Klaus Kaempf, para ajudar a acelerar as coisas. O primeiro objetivo é criar um pacote Debian do YaST 2 para facilitar o trabalho de desenvolvimento. Uma homepage está sendo montada para coordenar as atividades dos desenvolvedores no futuro.

O grupo de desenvolvimento de Mario Fux descobriu uma peculiaridade: internamente, o YaST 2 usa uma linguagem de programação proprietária, a *YaST Control Language* (YCP), para todos os módulos. Isso dificulta mais ainda o problema. Além de remover elementos específicos do SUSE do YaST 2, os desenvolvedores precisarão aprender uma nova linguagem de programação que nenhum outro projeto usa.

Embora nenhum outro programa de configuração facilite tanto a vida dos recém-chegados ao Linux quanto o YaST, usuários com conhecimentos especializados se queixam de que ele não lhes oferece margem suficiente para alterações manuais. Todavia, usuários de computadores de mesa normalmente não querem ter esse tipo de controle. Se o grupo for bem-sucedido em portar o conjunto completo de recursos, o YaST 2 pode alavancar a popularidade do Debian GNU/Linux como um sistema operacional para desktop. A única fonte de informações sobre o projeto de portagem no momento é a lista de discussão *Debian-Desktop* [2]. ■

Sistemas de arquivos no espaço do usuário

Se você já teve de mover um arquivo de um computador para outro através de uma rede, já está familiarizado com os problemas associados à transferência de arquivos: FTP é uma possibilidade, mas é inseguro. SCP é seguro, mas muito pouco amigável. Se for necessário apagar ou renomear um arquivo, você vai precisar de um login SSH.

Após abrir a sessão, pode-se rodar ferramentas familiares e convenientes como *cp*, *mv* e *rm*. Porém, SSH e SCP não permitem a execução remota desses comandos. Uma opção seria acrescentar comandos para apagar ou mover arquivos, como com o SCP. Porém, pelo lado ruim, os usuários precisariam decorar e digitar a sintaxe completa do comando para cada operação.

Miklos Szeredi parece ter se enchido disso e desenvolveu uma nova solução. Nos velhos tempos do LUFS (*Linux User File System*), ele usava o sistema de arquivos SSHFS, que simplesmente lhe permitia usar SSH como um driver de sistema de arquivos. Após montar o sistema de arquivos remoto usando SSHFS, os usuários tinham acesso a comandos típicos de shell. Infelizmente, o mantenedor do LUFS desistiu em pouco tempo e não se encontrou um sucessor. Como o Linux 2.6 se transformava rapidamente e o LUFS necessitava de um patch no kernel, o sistema era inútil.

Além disso, o projeto do LUFS tinha alguns pontos fracos que o tornavam difícil de manter. E ainda por cima, já havia uma alternativa a caminho: o *Fuse* (*Filesystem in Userspace*) é muito parecido com o LUFS do ponto de vista do usuário, embora use um método completamente diferente. Miklos Szeredi teve a idéia de reimplementar o LUFS-SSHFS baseado no Fuse e removendo os erros do projeto durante o processo. O resultado é também chamado de SSHFS.

E como o Fuse funciona? Em primeiro lugar, é preciso um kernel que permita o uso de Fuse. O Fuse não é um sistema de arquivos de verdade, mas uma estrutura que pode carregar drivers distintos do espaço do usuário. O driver do Fuse no kernel faz, portanto, a tradução entre o kernel e o driver do espaço do usuário. Superficialmente, o produto terminado tem a mesmíssima aparência de um sistema de arquivos normal; operações fundamentais, como ler, copiar ou apagar arquivos podem ser realizadas com as ferramentas normais do espaço do usuário. Uma das maiores vantagens do Fuse é que usuários sem privilégios de root podem montar o sistema de arquivos. Para isso, precisam carregar o módulo do Fuse no kernel.

Testado e aprovado

O SFTP também pode lidar com gerenciamento de arquivos em sistemas de arquivos remotos. Porém, como as operações ocorrem dentro de uma sessão do SFTP, o transporte de dados entre os sistemas remoto e local é mais complicado que com o SSHFS e as ferramentas padrão *cp*, *mv* e *rm*. Graças ao SSHFS, os usuários podem facilmente rodar scripts e outras ferramentas automátizadas no sistema remoto.

Outra vantagem do SSHFS é o suporte a *multi-threading*. Isso permite que tarefas múltiplas sejam realizadas consecutivamente. O SSHFS faz o trabalho mais leve, especialmente com tarefas de cópia recorrente que envolvam o SFTP ou o SCP.

A estrutura do Fuse é responsável principalmente por facilitar a integração de uma sessão SSH. Mas as opções são mais ou menos infinitas. Por exemplo, os usuários podem montar o serviço Gmail do Google via GmailFS, que também é baseado no Fuse, exatamente como um compartilhamento de arquivos por NFS. Dessa forma os usuários podem usar os dois gigabytes (desde primeiro de abril) de espaço oferecido pelo serviço para algo mais interessante. O desempenho do sistema de arquivos Fuse não chega àquilo que se esperaria de um sistema de arquivos de rede local ou com um fim especial; porém, desde que você tenha uma conexão rápida e estável, o Fuse pode ser uma opção útil. ■

Instalador do Debian e DevFS

A próxima versão do Debian GNU/Linux, codinome *Sarge*, não vai precisar de disquetes de boot para ser instalado. Em vez disso, conterá o novo *Debian Installer* (**figura 3**) com uma arquitetura completamente nova e muito mais robusta. Joey Hess publicou em janeiro deste ano a versão do instalador que será incluída no Sarge. Mas agora está parecendo que o aparentemente infinito processo de lançamento, que postergou o surgimento do Sarge por meses, começou a ficar sedento por vingança.

O principal problema é o *DevFS*. Essa solução para um diretório `/dev` organizado e desobstruído, que remonta aos tempos do Linux 2.4, caiu em desgraça junto aos desenvolvedores há um bom tempinho. Erros de projeto tornam o código difícil de manter. O Linux 2.6 tem uma nova abordagem: o *Udev*. Embora o DevFS ainda faça parte do kernel atual, há uma corrente que quer removê-lo para todo o sempre. Por exemplo, Linus Torvalds anunciou que ele seria removido do kernel em meados de 2005. Isso dará aos distribuidores uma oportunidade de substituir o antigo DevFS pelo UDev nesse meio tempo.

Isso afeta o instalador do Debian, que instalará o Linux 2.6 se necessário e habilitará o DevFS ao fazê-lo. Mudar isso exigiria um trabalho de muito baixo nível, que Joey Hass já declarou estar fora de questão. Isso significa mais uma trapalhada dos desenvolvedores do Debian: o Sarge será baseado num software obsoleto e não-suportado quando for finalmente

lançado. Embora isso pareça ser inofensivo à primeira vista – afinal de contas, as versões estáveis do Debian não usam o kernel mais recente – será preciso aplicar patches de desenvolvimento se ocorrerem problemas de segurança.

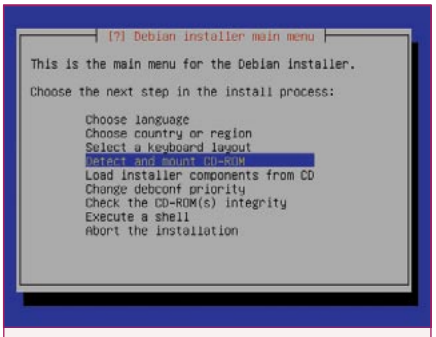

Figura 3: Problemas com o instalador do Debian: ele ainda usa o DevFS, um projeto órfão que deve ser removido do kernel num futuro próximo.

E esse é o maior risco do DevFS, já que não há mantenedor oficial para ele. Tanto as partes do DevFS no kernel quanto aquelas no espaço do usuário ficaram órfãs já há um bom tempo. Ou seja, a equipe de segurança do Debian precisaria de fato manter o DevFS para remover erros críticos ou remediar problemas de segurança. Isso significa um monte de trabalho, já que o Sarge terá muito mais pacotes do que a atual versão *Woody*; por isso, o fardo carregado pela Equipe de Segurança deverá aumentar consideravelmente. ■

O caminho para o Sarge...

Problemas desse tipo só podem ser evitados no futuro diminuindo-se drasticamente os intervalos entre as versões do Debian. Mesmo assim, não há qualquer sinal de um lançamento do Sarge, que está num estado mais ou menos indefinido há meses. Numa mensagem à lista de discussão *Debian-Devel-Announce* [2] há algumas semanas, Steve Langasek – membro da equipe de lançamento do Debian – anunciou que diversos problemas graves haviam sido resolvidos na versão de teste graças aos novos pacotes GCC, KDE 3.3 e Perl. Ele também mencionou muito trabalho duro para aumentar a segurança do Sarge e que havia hardware disponível para as arquiteturas permitidas.

...é longo e pedregoso

Isso é como uma luz no fim do túnel, uma vez que a falta de suporte à segurança tem sido vista como um enorme obstáculo nos últimos tempos. Para dar apoio a essa maré positiva, outra *Bug Squashing Party* (algo como *Festa do Inseticida*) foi realizada na primeira semana de fevereiro e os participantes foram bem-sucedidos em remover mais problemas. Se você está interessado em ler a mensagem de Langasek, na qual ele também comenta sobre as etapas futuras, dê uma olhada nos arquivos da lista de discussão em [3].

Os membros da equipe de segurança já estão habituados ao problema do software obsoleto. Está cada vez mais difícil encontrar patches de segurança para o Debian Woody. A versão atual tem mais de dois anos – e novos patches são com freqüência extremamente difíceis de instalar. Se forem precisos mais alguns anos até que surja o sucessor do Sarge, a questão é como a Equipe de Segurança poderá continuar a carregar o DevFS nas costas. ■

Isso é tu-tu-tudo, pe-pe-pessoal...

... ao menos neste mês. Se você quiser recomendar um programa que gostaria de ver nesta seção, que tal me mandar um email com sua sugestão [4]? Aguardo ansiosamente seus comentários! ■

Informações

[1] The Battle for Wesnoth: www.wesnoth.org

[2] Lista de discussão Debian-Desktop: lists.debian.org/debian-desktop

[3] Steve Langasek sobre atualizações no Sarge: lists.debian.org/debian-devel-announce/2005/01/msg00011.html

[4] Dicas e sugestões (em inglês): projects@linux-magazine.com