Guia de segurança do Debian

Javier Fernández-Sanguino Peña <jfs@debian.org>

Guia de segurança do Debian

por Javier Fernández-Sanguino Peña

Resumo

Este documento descreve a segurança no sistema Debian. Iniciando com o processo de tornar mais seguro e fortalecer a instalação padrão da distribuição Debian GNU/Linux. Ele também cobre algumas das tarefas mais comuns para configurar um ambiente de rede seguro usando a Debian GNU/Linux, oferece informações adicionais sobre as ferramentas de segurança disponíveis e fala sobre como a segurança é fornecida na Debian pelo time de segurança

Índice

Lista de Exemplos

Capítulo 1. Introdução

Uma das coisas mais difíceis sobre escrever documentos relacionado a segurança é que cada caso é único. Duas coisas que deve prestar atenção são o ambiente e as necessidades de segurança de um site, máquina ou rede. Por exemplo, a segurança necessária para um usuário doméstico é completamente diferente de uma rede em um banco. Enquanto a principal preocupação que um usuário doméstico tem é confrontar o tipo de cracker script kiddie, uma rede de banco tem preocupação com ataques diretos. Adicionalmente, o banco tem que proteger os dados de seus consumidores com precisão aritmética. Em resumo, cada usuário deve considerar o trajeto entre a usabilidade e paranóia/segurança.

Note que este manual somente cobre assuntos relacionados a software. O melhor software do mundo não pode te proteger se alguém puder ter acesso físico a máquina. Você pode coloca-la sob sua mesa, ou você pode coloca-la em um cofre fechado com uma arma de frente para ela. Não obstante a computação desktop pode ser muito mais segura (do ponto de vista do software) que uma fisicamente protegida caso o desktop seja configurado adequadamente e o programa na máquina protegida esteja cheio de buracos de segurança. Obviamente, você deverá considerar ambos os casos.

Este documento apenas lhe dará uma visão do que pode aumentar em segurança no sistema Debian GNU/Linux. Se ler outros documentos relacionados a segurança em Linux, você verá que existem assuntos comuns que se cruzarão com os citados neste documento. No entanto, este documento não tentará ser a última fonte de informações que deverá estar usando, ele tentará adaptar esta mesma informação de forma que seja útil no sistema Debian GNU/Linux. Distribuições diferentes fazem coisas de forma diferente (inicialização de daemons é um exemplo); aqui, você encontrará materiais que são apropriados para os procedimentos e ferramentas da Debian.

Autores

O mantenedor atual deste documento é mailto:jfs@debian.org. Por favor encaminhe a ele quaisquer comentários, adições e sugestões, e ele considerará a inclusão em lançamentos futuros deste manual.

This manual was started as a *HOWTO* by Alexander Reelsen. After it was published on the Internet, Javier Fernández-Sanguino Peña incorporated it into the Debian Documentation Project [http://www.debian.org/doc]. A number of people have contributed to this manual (all contributions are listed in the changelog) but the following deserve special mention since they have provided significant contributions (full sections, chapters or appendices):

- Stefano Canepa
- Era Eriksson
- Carlo Perassi
- Alexandre Ratti
- Jaime Robles
- Yotam Rubin
- Frederic Schutz
- Pedro Zorzenon Neto
- Oohara Yuuma
- Davor Ocelic

Where to get the manual (and available formats)

You can download or view the latest version of the Securing Debian Manual from the Debian Documentation Project [https://www.debian.org/doc/user-manuals#securing]. If you are reading a copy from another site, please check the primary copy in case it provides new information. If you are reading a translation, please review the version the translation refers to to the latest version available. If you find that the version is behind please consider using the original copy or review the to see what has changed.

If you want a full copy of the manual you can either download the text version [https://www.debian.org/doc/manuals/securing-debian-manual/securing-debian-manual.en.txt] or the PDF version [https://www.debian.org/doc/manuals/securing-debian-manual/securing-debian-manual.en.pdf] from the Debian Documentation Project's site. These versions might be more useful if you intend to copy the document over to a portable device for offline reading or you want to print it out. Be forewarned, the manual is over two hundred pages long and some of the code fragments, due to the formatting tools used, are not wrapped in the PDF version and might be printed incomplete.

The document is also provided in text, html and PDF formats in the harden-doc [http://packages.debian.org/harden-doc] package. Notice, however, that the package maybe not be completely up to date with the document provided on the Debian site (but you can always use the source package to build an updated version yourself).

This document is part of the documents distributed by the Debian Documentation Project [https://www.debian.org/doc/ddp]. You can review the changes introduced in the document using a web browser and obtaining information from the version control logs online [https://salsa.debian.org/ddp-team/securing-debian-manual]. You can also checkout the code using Git with the following call in the command line:

```
$ git clone https://salsa.debian.org/ddp-team/securing-debian-manual.git
```

Notas de organização/Retorno

Agora a parte oficial. No momento, eu (Alexandre Reelsen) escrevi a maioria dos parágrafos deste manual, mas em minha opinião este não deve ser o caso. Eu cresci e vivi com software livre, ele é parte do meu dia a dia e eu acho que do seu também. Eu encorajo a qualquer um para me enviar retorno, dicas, adições ou qualquer outra sugestão que possa ter.

Se achar que pode manter melhor uma certa seção ou parágrafo, então escreva um documento ao maintainer (mantenedor) e você será bem vindo a faze-lo. Especialmente se você encontrar uma seção marcada como FIXME, que significa que os autores não tem tempo ainda ou precisam de conhecimento sobre o tópico, envie um e-mail para eles imediatamente.

O tópico deste manual torna isto bastante claro que é importante mantê-lo atualizado, e você pode fazer sua parte. Por favor contribua.

Conhecimento necessário

A instalação do sistema Debian GNU/Linux não é muito difícil e você deverá ser capaz de instala-lo. Se você já tem algum conhecimento sobre o Linux ou outros tipo de Unix e você está um pouco familiar com a segurança básica, será fácil entender este manual, como este documento não explicará cada detalhe pequeno de características (caso contrário você terá um livro ao invés de um manual). Se não estiver

familiar, no entanto, você poderá dar uma olhada em “Conhecimento necessário” para ver onde achar informações atualizadas.

Coisas que precisam ser escritas (FIXME/ TODO)

Esta seção descreve todas as coisas que precisam ser corrigidas neste manual. Alguns parágrafos incluem as tags *FIXME* ou *TODO* descrevendo qual conteúdo esta faltando (ou que tipo de trabalho precisa ser feito). O propósito desta seção é descrever todas as coisas que precisam ser incluídas em um lançamento futuro do Manual, ou melhorias que precisam ser feitas (ou que são interessantes de serem adicionadas).

Se sente que pode fornecer ajuda contribuindo com a correção de conteúdo em qualquer elemento desta lista (ou anotações inline), contacte o autor principal (“Autores”

- This document has yet to be updated based on the latest Debian releases. The default configuration of some packages need to be adapted as they have been modified since this document was written.
- Expand the incident response information, maybe add some ideas derived from Red Hat's Security Guide's chapter on incident response [https://web.archive.org/web/20100412191348/http://www.redhat.com/docs/manuals/linux/RHL-9-Manual/security-guide/ch-response.html].
- Write about remote monitoring tools (to check for system availability) such as monit, daemontools and mon. See Sysamin Guide [https://web.archive.org/web/20100110040204/http://linuxdevcenter.com/pub/a/linux/2002/05/09/sysadminguide.html].
- Considere escrever uma seção sobre como fazer operações em rede com redes baseadas em sistemas Debian (com informações tal como o sistema básico, equivs e FAI).
- Check if this site [https://web.archive.org/web/20040731082209/http://www.giac.org/practical/gsec/Chris_Koutras_GSEC.pdf] has relevant info not yet covered here.
- Add information on how to set up a laptop with Debian, look here [https://web.archive.org/web/20040725013857/http://www.giac.org/practical/gcux/Stephanie_Thomas_GCUX.pdf].
- Adicione informações sobre como fazer um firewall usando o sistema Debian GNU/Linux. A seção relacionada a firewall é orientada atualmente sobre um sistema simples (não protegendo outros...) também fale sobre como testar a configuração.
- Add information on setting up a proxy firewall with Debian GNU/Linux stating specifically which packages provide proxy services (like xfwp, ftp-proxy, redir, smtpd, dnrd, jftpgw, oops, pdnsd, perdition, transproxy, tsocks). Should point to the manual for any other info. Note that zorp is now available as a Debian package and *is* a proxy firewall (they also provide Debian packages upstream).
- Informações sobre a configuração de serviços com o file-rc
- Verifique todas as URLs de referência e remova/corrija as que não estão mais disponíveis.
- Adicione informações sobre as substituições disponíveis (na Debian) para serviços padrões que são úteis para funcionalidades limitadas. Exemplos:
  - lpr local com o cups (pacote)?
  - lpr remota com o lpr
  - bind com dnrd/maradns

  - apache com dhttpd/thttpd/wn (tux?)
  - exim/sendmail com ssmtpd/smtpd/postfix
  - squid com tinyproxy
  - ftpd com oftpd/vsftp
  - ...
- Mais informações sobre patches do kernel relacionadas a segurança, incluindo os acima e informações específicas de como ativar estes patches em um sistema Debian.
  - Linux Intrusion Detection (kernel-patch-2.4-lids)
  - Linux Trustees (no pacote trustees)
  - NSA Enhanced Linux [http://wiki.debian.org/SELinux]
  - linux-patch-openswan
  - ...
- Detalhes sobre como desligar serviços desnecessários (como o **inetd**), é parte do procedimento de fortalecimento mas pode ser um pouco mais abrangente.
- Informações relacionadas a rotacionamento de senhas que é diretamente relacionada a política.
- Policy, e educação de usuários sobre a política.
- Mais sobre tcpwrappers, e wrappers em geral?
- O arquivo `hosts.equiv` e outros maiores buracos de segurança.
- Assuntos relacionados a serviços de compartilhamento de arquivos tais como Samba e NFS?
- suidmanager/dpkg-statoverrides.
- lpr e lprng.
- Desligar os ítens do gnome relacionados a IP
- Talk about pam_chroot (see http://lists.debian.org/debian-security/2002/05/msg00011.html) and its usefulness to limit users. Introduce information related to https://web.archive.org/web/20031204060940/http://www.securityfocus.com/infocus/1575. pdmenu, for example is available in Debian (whereas flash is not).
- Talk about chrooting services, some more info on this Linux Focus article [http://www.linuxfocus.org/English/January2002/article225.shtml].
- Fale sobre programas para fazer jaulas chroot. Compartment e chrootuid estão aguardando na incoming. Alguns outros como o (makejail, jailer) podem também serem introduzidos.
- Mais informações relacionadas a programas de análise de logs (i.e. logcheck e logcolorise).
- roteamento 'avançado' (policiamento de tráfego é relacionado a segurança)
- limitando o acesso do **ssh** a executar somente certos comandos.

- usando o dpkg-statoverride.
- métodos seguros de compartilhar um gravador de CD entre usuários.
- métodos seguros de fornecer som em rede em adição a características display (assim o som de clientes X são enviados para o hardware de som do servidor X).
- tornando navegadores mais seguros.
- setting up ftp over **ssh**.
- usando sistemas de arquivos loopback criptográficos.
- encrypting the entire file system.
- ferramentas de steganografia.
- ajustando um PKA para uma empresa.
- usando o LDAP para gerenciar usuários. Existe um howto do ldap+kerberos para o Debian em www.bayour.com escrito por Turbo Fredrikson.
- Como remover informações de utilidade reduzida em sistemas de produção tal como `/usr/share/doc`, `/usr/share/man` (sim, segurança pela obscuridade).
- Mais informações baseadas em ldap dos pacotes contendo os arquivos README (bem, não ainda, mas veja http://bugs.debian.org/cgi-bin/bugreport.cgi?bug=169465) e a partir do artigo LWN: http://lwn.net/1999/1202/kernel.php3.
- Add Colin's article on how to setup a chroot environment for a full sid system (https://web.archive.org/web/20030204012846/https://people.debian.org/~walters/chroot.html).
- Adicionar informações sobre como executar múltiplos sensores do snort em um determinado sistema (checar pelos relatórios de falhas enviados para o snort)
- Adicionar informações sobre como configurar um honeypot (honeyd)
- Describe situation wrt to FreeSwan (orphaned) and OpenSwan. VPN section needs to be rewritten.
- Add a specific section about databases, current installation defaults and how to secure access.
- Add a section about the usefulness of virtual servers (Xen et al).
- Explain how to use some integrity checkers (AIDE, integrit or samhain). The basics are simple and could even explain some configuration improvements.

Créditos e Agradecimentos!

- Alexander Reelsen escreveu o documento original.
- added more info to the original doc.
- Robert van der Meulen forneceu parágrafos relacionados a quota e muitas boas idéias.
- Ethan Benson corrigiu o parágrafo sobre PAM e adicionou boas idéias.
- Dariusz Puchalak contribuiu com algumas informações para vários capítulos.

- Gaby Schilders contribuiu com uma bela idéia geniosa/paranóica.
- Era Eriksson suavizou idiomas em vários lugares e contribuiu com o apêndice com a lista de checagens.
- Philipe Gaspar escreveu detalhes sobre LKM.
- Yotam Rubin contribuiu com correções para muitos erros assim como com informações relacionadas a versões do bind e senhas md5.
- Francois Bayart provided the appendix describing how to set up a bridge firewall.
- Joey Hess wrote the section describing how Secure Apt works on the Debian Wiki [http://wiki.debian.org/SecureApt].
- Martin F. Krafft wrote some information on his blog regarding fingerprint verification which was also reused for the Secure Apt section.
- Francesco Poli did an extensive review of the manual and provided quite a lot of bug reports and typo fixes which improved and helped update the document.
- Todas as pessoas que fizeram sugestões para melhorias que (eventualmente) serão incluídas aqui (veja “Where to get the manual (and available formats)”)
- (Alexander) todas as pessoas que me encorajaram a escrever este HOWTO (que mais adiante se tornou em um Manual).
- A todo o projeto Debian.

Capítulo 2. Antes de você iniciar

Para que finalidade você quer este sistema?

A segurança no Debian não é tão diferente da segurança em qualquer outro sistema; para implementar a segurança de maneira adequada, você deve primeiro decidir o que você pretende fazer com seu sistema. Após isto, você terá que considerar que as seguintes tarefas precisam ser executadas com cuidado se você realmente quer ter um sistema seguro.

Durante a leitura deste manual você verá tarefas para fazer antes, durante e após você instalar seu sistema Debian. As tarefas são ações como:

- Decidir quais serviços você necessita e limitar o sistema a eles. Isto inclui desativar/desinstalar serviços desnecessários e adicionar filtros como firewall ou tcpwrappers.
- Limitar usuários e permissões em seu sistema.
- Proteger os serviços oferecidos de modo que, em caso de problemas com um serviço, o impacto em seu sistema seja minimizado.
- Utilizar ferramentas apropriadas para garantir que o uso desautorizado seja detectado, de modo que você possa tomar as medidas apropriadas.

Esteja ciente dos problemas gerais de segurança

Este manual normalmente não entra em detalhes do "porque" algumas coisas são consideradas risco de segurança. Porém, você deve procurar algum conhecimento a mais sobre segurança em sistemas UNIX e em sistemas Linux especificamente. Reserve algum tempo para ler alguns documentos sobre segurança, de modo que você decida conscientemente quando se deparar com diferentes escolhas. O Debian é baseado no kernel do Linux, então você deve procurar muita informação sobre kernel Linux, Debian, outras distribuições e sobre segurança UNIX (mesmo que as ferramentas usadas ou os programas disponíveis sejam diferentes).

Alguns documentos úteis incluem:

- The http://www.tldp.org/HOWTO/Security-HOWTO/ is one of the best references regarding general Linux security.
- The http://www.tldp.org/HOWTO/Security-Quickstart-HOWTO/ is also a very good starting point for novice users (both to Linux and security).
- O http://seifried.org/lasg/ (fornecido no Debian através do pacote lasg) é um guia completo que aborda tudo relacionado a segurança Linux, da segurança do kernel até VPNs. É importante observar que este guia não é atualizado desde 2001, mas algumas informações ainda são relevantes. [1]
- O http://seifried.org/security/os/linux/20020324-securing-linux-step-by-step.html de Kurt Seifried.
- In http://www.tldp.org/links/p_books.html#securing_linux you can find a similar document to this manual but related to Red Hat, some of the issues are not distribution-specific and also apply to Debian.

[1] Por um tempo ele foi substituído pelo "Linux Security Knowledge Base". Esta documentação era fornecida no Debian através do pacote lskb. Agora ela voltou ao pacote *Lasg* novamente.

- Another Red Hat related document is https://web.archive.org/web/20050520170309/https://ltp.sourceforge.net/docs/RHEL-EAL3-Configuration-Guide.pdf.

- IntersectAlliance has published some documents that can be used as reference cards on how to harden Linux servers (and their services), the documents are available at https://web.archive.org/web/20030210231943/http://www.intersectalliance.com/projects/index.html.

- For network administrators, a good reference for building a secure network is the https://web.archive.org/web/20030418093551/http://www.linuxsecurity.com/docs/LDP/Securing-Domain-HOWTO/.

- Se você quer avaliar os programas que pretende usar (ou quer construir seus próprios programas) você deve ler o http://www.tldp.org/HOWTO/Secure-Programs-HOWTO/ (uma cópia está disponível em http://www.dwheeler.com/secure-programs/, ela inclui slides e comentários do autor, David Wheeler)

- Se você está considerando instalar um firewall, você deve ler o http://www.tldp.org/HOWTO/Firewall-HOWTO.html e o http://www.tldp.org/HOWTO/IPCHAINS-HOWTO.html (para kernels anteriores ao 2.4).

- Finally, a good card to keep handy is the https://web.archive.org/web/20030308013020/http://www.linuxsecurity.com/docs/QuickRefCard.pdf.

Em qualquer caso, existe mais informação sobre os serviços explanados aqui (NFS, NIS, SMB...) em muitos HOWTOs de http://www.tldp.org/. Alguns destes documentos falam em segurança relacionada a um determinado serviço, então certifique-se de procurar com cuidado.

The HOWTO documents from the Linux Documentation Project are available in Debian GNU/Linux through the installation of the doc-linux-text (text version) or doc-linux-html (HTML version). After installation these documents will be available at the `/usr/share/doc/HOWTO/en-txt` and `/usr/share/doc/HOWTO/en-html` directories, respectively.

Outros livros sobre Linux recomendados:

- Maximum Linux Security : A Hacker's Guide to Protecting Your Linux Server and Network. Anônimo. Paperback - 829 páginas. Sams Publishing. ISBN: 0672313413. Julho 1999.

- Linux Security By John S. Flowers. New Riders; ISBN: 0735700354. Março 1999

- https://web.archive.org/web/20030202131658/https://www.linux.org/books/ISBN_0072127732.html By Brian Hatch. McGraw-Hill Higher Education. ISBN 0072127732. April, 2001

Outros livros (que podem ser relacionados a assuntos sobre UNIX e segurança e não especificamente sobre Linux):

- https://web.archive.org/web/20030206231652/http://www.oreilly.com/catalog/puis/ Garfinkel, Simpson, and Spafford, Gene; O'Reilly Associates; ISBN 0-56592-148-8; 1004pp; 1996.

- Firewalls and Internet Security Cheswick, William R. and Bellovin, Steven M.; Addison-Wesley; 1994; ISBN 0-201-63357-4; 320pp.

Alguns Web sites úteis sobre segurança para manter-se atualizado:

- http://csrc.nist.gov/.

- https://cve.mitre.org/data/refs/refmap/source-BUGTRAQ.html CVE Reference Map for Source BUGTRAQ

- http://www.linuxsecurity.com/. General information regarding Linux security (tools, news...). Most useful is the https://linuxsecurity.com/howtos page.

Como o Debian controla a segurança do sistema?

Agora que você tem uma visão geral da segurança no Debian GNU/Linux observe mais algumas considerações para construir um sistema seguro:

- Debian problems are always handled openly, even security related. Security issues are discussed openly on the debian-security mailing list. Debian Security Advisories (DSAs) are sent to public mailing lists (both internal and external) and are published on the public server. As the http://www.debian.org/social_contract states: *We will not hide problems. We will keep our entire bug report database open for public view at all times. Reports that people file online will promptly become visible to others.*

- O Debian sempre procura corrigir os problemas de segurança. A equipe de segurança verifica muitas fontes relacionadas a segurança, a mais importante sendo http://www.securityfocus.com/cgi-bin/vulns.pl, sempre a procura de pacotes que aumentem a segurança e que possam ser incluídos.

- Atualizações de seguranaça estão em primeira prioridade. Quando um problema aparece em um pacote Debian, a atualização de segurança é preparada o mais rápido possível e incluída nas distribuições estável e instável para todas as arquiteturas.

- Informações sobre segurança estão centralizadas em http://security.debian.org/.

- O Debian está sempre tentando aumentar sua segurança através de novos projetos como o mecanismo automático de verificação de assinatura dos pacotes.

- O Debian fornece um grande número de ferramentas de segurança para administração de monitoramento do sistema. Desenvolvedores tentam integrar estas ferramentas com a distribuição para fazer um sistema opeacional cada vez mais seguro. Estas ferramentas incluem: verificação da integridade do sistema, firewall, detecção de intrusos, etc.

- Mantenedores de pacote estão cientes dos problemas de segurança. Isto leva a pensar que algumas restrições poderiam ser impostas para alguns serviços em seu uso normal. O Debian, porém, tenta balancear segurança e facilidade de administração - os programas não são desativados quando você os instala (como é o caso nas distribuições da família BSD). Em qualquer caso, implementações de segurança tipo programas `setuid` são parte da política http://www.debian.org/doc/debian-policy/.

Publicando informações de segurança específica para o Debian e complementando outros documentos relacionados a segurança (veja “Conhecimento necessário”), este documento ajuda a produzir sistemas mais seguros.

Capítulo 3. Antes e durante a instalação

Escolha uma senha para a BIOS

Antes de instalar qualquer sistema operacional em seu computador, configure uma senha para a BIOS. Após a instalação (uma vez que você tenha habilitado o boot a partir do HD) você deve voltar a BIOS e alterar a sequência de boot desabilitando o boot a partir do disquete (floppy), cdrom e outros dispositivos. Se você não fizer assim, um cracker só precisará de acesso físico e um disco de boot para acessar o sistema inteiro.

Desabilitar o boot a menos que uma senha seja fornecida é bem melhor. Isto pode ser muito eficaz num servidor, porque ele não é reiniciado constantemente. A desvantagem desta tática é que o reinício exige intervenção humana, o que pode causar problemas se a máquina não for facilmente acessível.

Observação: muitas BIOS vem de fábrica com senhas padrão bem conhecidas e existem programas que recuperam estas senhas, ou seja, alteram a senha atual para a senha original, para o caso de uma perda da senha pelo administrador. Assim, não dependa desta medida para proteger o acesso ao sistema.

Particionando o sistema

Escolha um esquema de partição inteligente

Um esquema de partição inteligente depende de como a máquina será usada. Uma boa regra é ser razoavelmente generoso com suas partições e prestar atenção aos seguintes fatores:

- Qualquer diretório que um usuário tenha permissões de escrita, como o `/home`, `/tmp` e o `/var/tmp/`, devem estar separados em uma partição. Isto reduz o risco de um usuário malicioso utilizar o DoS (Denial of Service) para encher seu diretório raiz (/) e tornar o sistema inutilizável (Observação: isto não é totalmente verdade uma vez que sempre existe algum espaço reservado para o usuário root que o usuário normal não pode preencher), e também previne ataques tipo hardlink. [1]

- Qualquer partição com dados variáveis, isto é, `/var` (especialmente `/var/log`) também deve estar numa partição separada. Em um sistema Debian você deve criar `/var` um pouco maior que em outros sistemas porque o download de pacotes (cache do apt) é armazenado em `/var/cache/apt/archives`.

- Qualquer partição onde você queira instalar software que não é padrão da distribuição deve estar separada. De acordo com a Hierarquia Padrão do Sistema de Arquivos, estas são `/opt` ou `/usr/local`. Se estas partições estão separadas, elas não serão apagadas se você (tiver que) reinstalar o Debian.

- Do ponto de vista da segurança, é sensato tentar mover os dados estáticos para sua própria partição e então montar esta partição somente para leitura. Melhor ainda será colocar os dados numa mídia somenta para leitura. Veja abaixo para mais detalhes.

[1] Um bom exemplo deste tipo de ataque usando /tmp é detalhado em http://www.hackinglinuxexposed.com/articles/20031111.html e http://www.hackinglinuxexposed.com/articles/20031214.html (Observe que o incidente é um relato Debian) Ele é basicamente um ataque no qual um usuário local usa uma aplicação setuid vulnerável através de um hard link para ela analisando qualque atualização (ou remoção) do próprio binário feita pelo administrador do sistema. Dpkg foi recentemente corrigido para prevenir isto (veja http://bugs.debian.org/225692) mas outros binários setuid (não controlados pelo gerenciador de pacotes) correm o risco se as partições não estiverem configuradas corretamente.

No caso de um servidor de email é importante ter uma partição separada para o spool de email. Usuários remotos (conhecidos ou não) podem encher o spool de email (`/var/mail` e/ou `/var/spool/mail`). Se o spool está em uma partição separada, esta situação não tornará o sistema inutilizável. Porém (se o diretório de spool está na mesma partição que `/var`) o sistema pode ter sérios problemas: log não serão criados, pacotes podem não ser instalados e alguns programas podem ter problemas ao iniciar (se eles usam `/var/run`).

Para partições que você não tem certeza do espaço necesário, você pode instalar o Logical Volume Manager (lvm-common e os binários necessário para o kernel, estes podem ser lvm10, lvm6, ou lvm5). Usando `lvm`, você pode criar grupos de volume que expandem múltiplos volumes físicos.

Escolhendo o sistema de arquivos apropriado

During the system partitioning you also have to decide which file system you want to use. The default file system[2] selected in the Debian installation for Linux partitions is `ext3`, a journaling file system. It is recommended that you always use a journaling file system, such as `ext3`, `reiserfs`, `jfs` or `xfs`, to minimize the problems derived from a system crash in the following cases:

- Para laptops em todos os sistemas de arquivos instalados. Assim se acabar a bateria inesperadamente ou o sistema congelar você correrá menos risco de perda de dados durante a reinicialização do sistema.
- para sistemas que armazenam grande quantidade de dados (como servidores de email, servidores ftp, sistemas de arquivos de rede ....). Assim, em caso de queda, menos tempo será gasto para o servidor checar o sistema de arquivos e a probabilidade da perda de dados será menor.

Deixando de lado a performance dos sistemas journalling (uma vez que isto pode iniciar uma verdadeira guerra), normalmente é melhor usar o `ext3`. A razão para isto é que ele é compatível com o antigo `ext2`, assim se existe alguma parte do seu sistema com journalling você pode desabilitar este recurso e ainda ter um sistema em condições de trabalhar. Também, se você precisar recuperar o sistema com um disco de boot (ou CDROM) você não precisa personalizar o kernel. Se o kernel é 2.4, o suporte a `ext3` já está disponível, se é um kernel 2.2 você será capaz de iniciar o sistema de arquivos mesmo se perder as capacidades journalling. Se você estiver usando outro sisteama journalling diferente do `ext3`, você pode não ser capaz de recuperar o sistema a menos que você tenha um kernal 2.4 com os módulos necessários instalados. Se seu disco de resgate tem o kernel 2.2 pode ser mais difícil acessar sistemas `reiserfs` ou `xfs`.

Em qualquer caso, a integridade dos dados pode ser melhor usando `ext3` uma vez que ele usa file-data journalling enquanto outros usam apenas meta-data journalling, veja http://lwn.net/2001/0802/a/ext3-modes.php3.

Notice, however, that there are some partitions that might not benefit from using a journaling filesystem. For example, if you are using a separate partition for `/tmp/` you might be better off using a standard `ext2` filesystem as it will be cleaned up when the system boots.

Não conecte-se a internet até estar pronto

O sistema não deve ser imediatamente conectado a internet durante a instalação. Isto pode parecer estúpido mas intalação via internet é um método comum. Uma vez que o sistema instalará e ativará serviços imediatamente, se o sistema estiver conectado a internet e os serviços não estiverem adequadamente configurados, você estará abrindo brechas para ataques.

Observe também que alguns serviços podem ter vulnerabilidades de segurança não corrigidas nos pacotes que você estiver usando para a instalação. Isto normalmente será verdade se você estiver instalando a partir

[2] Since Debian GNU/Linux 4.0, codename `etch`

de mídia antiga (como CD-ROMs). Neste caso, o sistema poderia estar comprometido antes de terminar a instalação!

Uma vez que a instalação e atualizações do Debian podem ser feitas pela internet você pode pensar que é uma boa idéia usar este recurso na instalação. Se o sistema está diretamente conectado (e não está protegido por um firewall ou NAT), é melhor instalar sem conexão com a grande rede usando um mirror local com os pacotes do Debian e as atualizações de segurança. Você pode configurar mirrors de pacotes usando outro sistema conectado com ferramentas específicas do Debian (se ele é um sistema tipo Debian) como apt-move ou apt-proxy, ou outras, para fornecer os arquivos para o sistema instalado. Se não puder fazer isto, você pode configurar regras de firewall para limitar o acesso ao sistema enquanto estiver atualizando (veja “Atualização de segurança protegida por um firewall”).

Configure a senha do root

Configurar uma boa senha para o root é o requerimento mais básico para ter um sistema seguro. Veja passwd(1) para mais dicas de como criar boas senhas. Você também pode usar um programa gerador de senhas para fazer isto para você (veja “Gerando senhas de usuários”).

Plenty of information on choosing good passwords can be found on the Internet; two that provide a decent summary and rationale are Eric Wolfram's http://wolfram.org/writing/howto/password.html and Walter Belgers' https://web.archive.org/web/20030218000949/http://www.belgers.com/write/pwseceng.txt

Rode o mínimo de serviços necessários

Serviços são programas como servidores ftp e servidores web. Uma vez que eles tem que estar *escutando* por conexões que requisitem o serviço, computadores externos podem conectar-se a eles. Serviços algumas vezes são vulneráveis (i.e. podem estar comprometidos sobre um certo ataque) e oferecem risco a segurança.

Você não deve instalar serviços que não são necessários em sua máquina. Todo serviço instalado pode introduzir novos, talvez não óbvios ou conhecidos, buracos de segurança em seu computador.

As you may already know, when you install a given service the default behavior is to activate it. In a default Debian installation, with no services installed, the number of running services is quite low and the number of network-oriented services is even lower. In a default Debian 3.1 standard installation you will end up with OpenSSH, Exim (depending on how you configured it) and the RPC portmapper available as network services[3]. If you did not go through a standard installation but selected an expert installation you can end up with no active network services. The RPC portmapper is installed by default because it is needed for many services, for example NFS, to run on a given system. However, it can be easily removed, see “Tornando serviços RPC mais seguros” for more information on how to secure or disable RPC services.

Quando você instala um novo serviço de rede (daemon) em seu sistema Debian GNU/Linux ele pode ser habilitado de duas maneiras: através do superdaemon **inetd** (uma linha será adicionada ao `/etc/inetd.conf`) ou através de um programa que serve de interface. Estes programas são controlados pelos arquivos `/etc/init.d`, que são chamados no momento da inicialização através do mecanismo SysV (ou outro alternativo) pelo uso de symlinks em `/etc/rc?.d/*` (para mais informações de como isto é feito leia `/usr/share/doc/sysvinit/README.runlevels.gz`).

If you want to keep some services but use them rarely, use the **update-*** commands, e.g. **update-inetd** and **update-rc.d** to remove them from the startup process. For more information on how to disable network services read “Desabilitando daemons de serviço”. If you want to change the default behaviour of starting

[3] The footprint in Debian 3.0 and earlier releases wasn't as tight, since some **inetd** services were enabled by default. Also standard installations of Debian 2.2 installed the NFS server as well as the telnet server.

up services on installation of their associated packages[4] use **policy-rc.d**, please read `/usr/share/doc/sysv-rc/README.policy-rc.d.gz` for more information.

invoke-rc.d support is mandatory in Debian, which means that for Debian 4.0 *etch* and later releases you can write a policy-rc.d file that forbids starting new daemons before you configure them. Although no such scripts are packaged yet, they are quite simple to write. See policyrcd-script-zg2.

Desabilitando daemons de serviço

Disabling a daemon service is quite simple. You either remove the package providing the program for that service or you remove or rename the startup links under `/etc/rc${runlevel}.d/`. If you rename them make sure they do not begin with 'S' so that they don't get started by **/etc/init.d/rc**. Do not remove all the available links or the package management system will regenerate them on package upgrades, make sure you leave at least one link (typically a 'K', i.e. kill, link). For more information read http://www.debian.org/doc/manuals/reference/ch-system.en.html#s-custombootscripts section of the Debian Reference (Chapter 2 - Debian fundamentals).

You can remove these links manually or using `update-rc.d` (see update-rc.d(8)). For example, you can disable a service from executing in the multi-user runlevels by doing:

```
# update-rc.d name stop XX 2 3 4 5 .
```

Observe que, se você *não* está usando file-rc, `update-rc.d -f _service_ remove` não trabalhará apropriadamente, pois embora *todos* links sejam removidos, após reinstalação ou upgrade do pacote estes links serão regenerados (provavelmente não é o que você quer). Se pensa que isto não é intuitivo você provavelmente está certo (veja http://bugs.debian.org/67095). Texto da manpage:

```
Se qualquer arquivo /etc/rcrunlevel.d/[SK]??name já existe então
update-rc.d não faz nada. É desta maneira que o administrador do sistema pode
reorgananizar os links, contanto que eles deixem pelo menos um link remanescente
sem ter sua configuração reescrita.
```

Se você está usando file-rc, toda informação sobre serviços é manipulada por um arquivo de configuração comum e é mantida mesmo se os pacotes forem removidos do sistema.

You can use the TUI (Text User Interface) provided by sysv-rc-conf to do all these changes easily (**sysv-rc-conf** works both for file-rc and normal System V runlevels). You will also find similar GUIs for desktop systems. You can also use the command line interface of sysv-rc-conf:

```
# sysv-rc-conf foobar off
```

The advantage of using this utility is that the rc.d links are returned to the status they had before the 'off' call if you re-enable the service with:

```
# sysv-rc-conf foobar on
```

Other (less recommended) methods of disabling services are:

- Removing the `/etc/init.d/service_name` script and removing the startup links using:

[4] This is desirable if you are setting up a development chroot, for example.

```
# update-rc.d name remove
```

- Move the script file (`/etc/init.d/service_name`) to another name (for example `/etc/init.d/OFF.service_name`). This will leave dangling symlinks under `/etc/rc${runlevel}.d/` and will generate error messages when booting up the system.
- Remove the execute permission from the `/etc/init.d/service_name` file. That will also generate error messages when booting.
- Edit the `/etc/init.d/service_name` script to have it stop immediately once it is executed (by adding an **exit 0** line at the beginning or commenting out the `start-stop-daemon` part in it). If you do this, you will not be able to use the script to startup the service manually later on.

Nevertheless, the files under `/etc/init.d` are configuration files and should not get overwritten due to package upgrades if you have made local changes to them.

Infelizmente, diferente de outros sistemas operacionanais tipo UNIX, os serviços no Debian não podem ser desabilitados pela modificação dos arquivos em `/etc/default/_servicename_`.

FIXME: Adicione mais informação sobre manipulação de daemons usando file-rc

Desabilitando o inetd ou seus serviços

Você deve checar se realmente precisa do daemon **inetd**. Inetd sempre foi uma maneira de compensar deficiências do kernel, mas estas deficiências foram corrigidas. Existe possibilidade de ataques DoS (Denial of Service) contra o **inetd**, então é preferível usar daemons individuais do que rodar um serviço do **inetd**. Se você ainda quer rodar algum serviço do **inetd**, então no mínimo alterne para um daemon mais configurável como **xinetd**, **rlinetd** ou **openbsd-inetd**.

You should stop all unneeded Inetd services on your system, like **echo**, **chargen**, **discard**, **daytime**, **time**, **talk**, **ntalk** and r-services (**rsh**, **rlogin** and **rcp**) which are considered HIGHLY insecure (use **ssh** instead).

Você pode desabilitar os serviços editando o arquivo `/etc/inetd.conf` diretamente, mas o Debian fornece uma alternativa melhor: `update-inetd` (o qual comenta os serviços de modo que eles possam facilmente ser reativados). Você pode remover o daemon **telnet** para alterar o arquivo de configuração e reiniciar o daemon (neste caso o serviço **telnet** é desabilitado):

```
/usr/sbin/update-inetd --disable telnet
```

Se você quer um serviço, mas não o quer disponível para todos os IP do seu host, você deve usar um recurso não documentado no **inetd** (substitua o nome do serviço por serviço@ip) ou use um daemon alternativo como **xinetd**.

Instale o mínimo de software necessário

O Debian vem com *uma grande quantidade* de software, por exemplo o Debian 3.0 *woody* inclui quase 6 CD-ROMs de software e milhares de pacotes. Apesar da grande quantidade de software, a instalação do sistema base utiliza poucos pacotes. [5] você pode estar mal informado e instalar mais que o realmente necessário para seu sistema.

Sabendo o que seu sistema realmente precisa, você deve instalar apenas o que for realmente necessário para seu trabalho. Qualquer ferramenta desnecessária pode ser usada por um usuário malicioso para

[5] Por exemplo, no Debian Woody ela gira em torno de 40Mbs, tente isto para ver quanto os pacotes necessários ocupam no sistema:

```
$ size=0 $ for i in `grep -A 1 -B 1 "^Section: base" /var/lib/dpkg/available | grep -A 2 "^Priority: required" |
```

comprometer o sistema ou por um invasor externo que tenha acesso ao shell (ou código remoto através de serviços exploráveis).

A presença, por exemplo, de utilitários de desenvolvimento (um compilador C) ou linguagens interpretadas (como **perl**, **python**, **tcl**...) pode ajudar um atacante a comprometer o sistema da seguinte maneira:

- permitir a ele fazer escalação de privilégios. Isto facilita, por exemplo, rodar exploits locais no sistema se existe um depurador e compilador prontos para compilar e testar.
- fornecer ferramentas que poderiam ajudar um atacante a usar o sistema comprometido como *base de ataque* contra outros sistemas [6]

É claro que um invasor com acesso ao shell local pode baixar suas próprias ferramentas e executá-las, além disso o próprio shell pode ser usado para fazer complexos programas. Remover software desnecessário não impedirá o problema mas dificultará a ação de um possível atacante. Então, se você deixar disponíveis ferramentas em um sistema de produção que poderiam ser usadas remotamente para um ataque (veja "Ferramentas de verificação remota de vulnerabilidades"), pode acontecer de um invasor usá-las.

Please notice that a default installation of Debian *sarge* (i.e. an installation where no individual packages are selected) will install a number of development packages that are not usually needed. This is because some development packages are of *Standard* priority. If you are not going to do any development you can safely remove the following packages from your system, which will also help free up some space:

```
Package                    Size
-------------------------+--------
gdb                     2,766,822
gcc-3.3                 1,570,284
dpkg-dev                  166,800
libc6-dev               2,531,564
cpp-3.3                 1,391,346
manpages-dev            1,081,408
flex                      257,678
g++                         1,384 (Note: virtual package)
linux-kernel-headers    1,377,022
bin86                      82,090
cpp                        29,446
gcc                         4,896 (Note: virtual package)
g++-3.3                 1,778,880
bison                     702,830
make                      366,138
libstdc++5-3.3-dev        774,982
```

This is something that is fixed in releases post-sarge, see http://bugs.debian.org/cgi-bin/bugreport.cgi?bug=301273 and http://bugs.debian.org/cgi-bin/bugreport.cgi?bug=301138. Due to a bug in the installation system this did not happen when installing with the installation system of the Debian 3.0 *woody* release.

Removendo Perl

Remover o **perl** pode não ser fácil em um sistema Debian pois ele é muito usado. O pacote perl-base tem prioridade classificada como requerida (*Priority: required*), o que já diz tudo. Você pode removê-lo mas

[6] Muitas invasões são feitas mais para acessar os recursos e executar atividades ilícitas (ataques denial of service, spam, rogue ftp servers, poluição dns...) do que para obter dados confidenciais dos sistemas comprometidos.

não será capaz de rodar qualquer aplicação **perl** no sistema; você ainda terá que enganar o sistema de gerenciamento de pacotes para ele pensar que o perl-base ainda está instalado. [7]

Quais utilitários usam **perl**? Você mesmo pode verificar:

```
$ for i in /bin/* /sbin/* /usr/bin/* /usr/sbin/*; do [ -f $i ] && {
type=`file $i | grep -il perl`; [ -n "$type" ] && echo $i; }; done
```

Estes incluem os seguintes utilitários em pacotes com prioridade *required* ou *important*:

- `/usr/bin/chkdupexe` do pacote util-linux.
- `/usr/bin/replay` of package bsdutils.
- `/usr/sbin/cleanup-info` do pacote dpkg.
- `/usr/sbin/dpkg-divert` do pacote dpkg.
- `/usr/sbin/dpkg-statoverride` do pacote dpkg.
- `/usr/sbin/install-info` do pacote dpkg.
- `/usr/sbin/update-alternatives` do pacote dpkg.
- `/usr/sbin/update-rc.d` do pacote sysvinit.
- `/usr/bin/grog` of package groff-base.
- `/usr/sbin/adduser` of package adduser.
- `/usr/sbin/debconf-show` do pacote debconf.
- `/usr/sbin/deluser` of package adduser.
- `/usr/sbin/dpkg-preconfigure` do pacote debconf.
- `/usr/sbin/dpkg-reconfigure` do pacote debconf.
- `/usr/sbin/exigrep` of package exim.
- `/usr/sbin/eximconfig` of package exim.
- `/usr/sbin/eximstats` of package exim.
- `/usr/sbin/exim-upgrade-to-r3` do pacote exim.
- `/usr/sbin/exiqsumm` of package exim.
- `/usr/sbin/keytab-lilo` do pacote lilo.
- `/usr/sbin/liloconfig` of package lilo.
- `/usr/sbin/lilo_find_mbr` do pacote lilo.
- `/usr/sbin/syslogd-listfiles` do pacote sysklogd.

[7] Você pode fazer (em outro sistema) um pacote dummy com o equivs

- `/usr/sbin/syslog-facility` do pacote sysklogd.
- `/usr/sbin/update-inetd` do pacote netbase.

Assim, sem Perl e, a menos que você recompile estes utilitários em um script shell, você provavelmente não será capaz de gerenciar nenhum pacote (assim você também não será capaza de atualizar o sistema, o que *não é uma coisa boa*).

Se você está determinado a remover o Perl do Debian e tem tempo de sobra, envie os relatórios de bugs referentes aos pacotes acima referidos incluindo possíveis substituições para os utilitários escritas em shell.

If you wish to check out which Debian packages depend on Perl you can use

```
$ grep-available -s Package,Priority -F Depends perl
```

or

```
$ apt-cache rdepends perl
```

Leia as listas de segurança do Debian (security mailing lists)

Nunca é demais dar uma olhada na lista debian-security-announce, onde avisos e correções dos pacotes são anunciadas pela equipe de segurança do Debian, ou na mailto:debian-security@lists.debian.org, onde você pode participar de discussões sobre assuntos relacionados a segurança Debian.

Para receber importantes atualizações de segurança e alertas envie email para mailto:debian-security-announce-request@lists.debian.org com a palavra "subscribe" como assunto. Você também pode inscrever-se nesta lista no endereço http://www.debian.org/MailingLists/subscribe

Esta lista tem pouco volume de mensagens e assinando ela você será imediatamente alertado sobre atualizações de segurança para a distribuição Debian. Isto lhe permitirá rapidamente baixar os novos pacotes com atualizações de segurança, as quais são muito importantes na manutenção de um sistema seguro. (Veja “Executar uma atualização de segurança” para detalhes de como fazer isto.)

Capítulo 4. Após a instalação

Once the system is installed you can still do more to secure the system; some of the steps described in this chapter can be taken. Of course this really depends on your setup but for physical access prevention you should read “Altere a BIOS (de novo)”,“Configurar a senha do LILO ou GRUB”, “Remover o aviso de root do kernel”, “Restringindo o acesso de login no console”, and “Restringindo reinicializações do sistema através da console”.

Before connecting to any network, especially if it's a public one you should, at the very least, execute a security update (see “Executar uma atualização de segurança”). Optionally, you could take a snapshot of your system (see “Fazendo um snapshot do sistema”).

Inscreva-se na lista de discussão "Anúncios de Segurança do Debian"

In order to receive information on available security updates you should subscribe yourself to the debian-security-announce mailing list in order to receive the Debian Security Advisories (DSAs). See “O time Debian Security” for more information on how the Debian security team works. For information on how to subscribe to the Debian mailing lists read http://lists.debian.org.

Os DSAs são assinados pelo time de segurança do Debian e as assinaturas podem ser pegas através do endereço http://security.debian.org.

You should consider, also, subscribing to the http://lists.debian.org/debian-security for general discussion on security issues in the Debian operating system. You will be able to contact other fellow system administrators in the list as well as Debian developers and upstream developers of security tools who can answer your questions and offer advice.

FIXME: também adicionar a chave aqui?

Executar uma atualização de segurança

As soon as new security bugs are detected in packages, Debian maintainers and upstream authors generally patch them within days or even hours. After the bug is fixed, a new package is provided on http://security.debian.org.

If you are installing a Debian release you must take into account that since the release was made there might have been security updates after it has been determined that a given package is vulnerable. Also, there might have been minor releases (there have been four for the Debian 3.0 *sarge* release) which include these package updates.

During installation security updates are configured for your system and pending updates downloaded and applied, unless you specifically opt out of this or the system was not connected to the Internet. The updates are applied even before the first boot, so the new system starts its life as up to date as possible.

To manually update the system, put the following line in your `sources.list` and you will get security updates automatically, whenever you update your system. Replace *`[CODENAME]`* with the release codename, e.g. *squeeze*.

```
deb http://security.debian.org/ stable/updates main contrib non-free
```

Note: If you are using the *testing* branch use the security testing mirror sources as described in “Security support for the testing branch”.

Once you've done this you can use multiple tools to upgrade your system. If you are running a desktop system you will have[1] an application called **update-notifier** that will make it easy to check if new updates are available, by selecting it you can make a system upgrade from the desktop (using **update-manager**). For more information see “Checking for updates at the Desktop”. In desktop environments you can also use synaptic (GNOME), kpackage or adept (KDE) for more advanced interfaces. If you are running a text-only terminal you can use aptitude, apt or dselect (deprecated) to upgrade:

- If you want to use aptitude's text interface you just have to press *u* (update) followed by *g* (to upgrade). Or just do the following from the command line (as root):

```
# aptitude update
# aptitude upgrade
```

- If you want to use apt do just like with aptitude but substitute the **aptitude** lines above with **apt-get**.
- If you want to use dselect then first [U]pdate, then [I]nstall and finally, [C]onfigure the installed/upgraded packages.

If you like, you can add the deb-src lines to `/etc/apt/sources.list` as well. See apt(8) for further details.

Security update of libraries

Once you have executed a security update you might need to restart some of the system services. If you do not do this, some services might still be vulnerable after a security upgrade. The reason for this is that daemons that are running before an upgrade might still be using the old libraries before the upgrade [2].

From Debian *Jessie* and up, you can install the needrestart package, which will run automatically after each APT upgrade and prompt you to restart services that are affected by the just-installed updates. In earlier releases, you can run the **checkrestart** program (available in the debian-goodies package) manually after your APT upgrade.

Some packages (like libc6) will do this check in the postinst phase for a limited set of services specially since an upgrade of essential libraries might break some applications (until restarted)[3].

Bringing the system to run level 1 (single user) and then back to run level 3 (multi user) should take care of the restart of most (if not all) system services. But this is not an option if you are executing the security upgrade from a remote connection (like ssh) since it will be severed.

Excercise caution when dealing with security upgrades if you are doing them over a remote connection like ssh. A suggested procedure for a security upgrade that involves a service restart is to restart the SSH daemon and then, immediately, attempt a new ssh connection without breaking the previous one. If the connection fails, revert the upgrade and investigate the issue.

[1] In *Etch* and later releases
[2] Even though the libraries have been removed from the filesystem the inodes will not be cleared up until no program has an open file descriptor pointing to them.
[3] This happened, for example, in the upgrade from libc6 2.2.x to 2.3.x due to NSS authentication issues, see http://lists.debian.org/debian-glibc/2003/03/msg00276.html.

Security update of the kernel

First, make sure your kernel is being managed through the packaging system. If you have installed using the installation system from Debian 3.0 or previous releases, your kernel is *not* integrated into the packaging system and might be out of date. You can easily confirm this by running:

```
$ dpkg -S `readlink -f /vmlinuz`
linux-image-2.6.18-4-686: /boot/vmlinuz-2.6.18-4-686
```

If your kernel is not being managed you will see a message saying that the package manager did not find the file associated to any package instead of the message above, which says that the file associated to the current running kernel is being provided by the linux-image-2.6.18-4-686. So first, you will need to manually install a kernel image package. The exact kernel image you need to install depends on your architecture and your prefered kernel version. Once this is done, you will be able to manage the security updates of the kernel just like those of any other package. In any case, notice that the kernel updates will *only* be done for kernel updates of the same kernel version you are using, that is, **apt** will not automatically upgrade your kernel from the 2.4 release to the 2.6 release (or from the 2.4.26 release to the 2.4.27 release[4]).

The installation system of recent Debian releases will handle the selected kernel as part of the package system. You can review which kernels you have installed by running:

```
$ COLUMNS=150 dpkg -l 'linux-image*' | awk '$1 ~ /ii/ { print $0 }'
```

To see if your kernel needs to be updated run:

```
$ kernfile=`readlink -f /vmlinuz`
$ kernel=`dpkg -S $kernfile | awk -F : '{print $1}'`
$ apt-cache policy $kernel
linux-image-2.6.18-4-686:
  Installed: 2.6.18.dfsg.1-12
  Candidate: 2.6.18.dfsg.1-12
  Version table:
 *** 2.6.18.dfsg.1-12 0
        100 /var/lib/dpkg/status
```

If you are doing a security update which includes the kernel image you *need* to reboot the system in order for the security update to be useful. Otherwise, you will still be running the old (and vulnerable) kernel image.

If you need to do a system reboot (because of a kernel upgrade) you should make sure that the kernel will boot up correctly and network connectivity will be restored, specially if the security upgrade is done over a remote connection like ssh. For the former you can configure your boot loader to reboot to the original kernel in the event of a failure (for more detailed information read Remotely rebooting Debian GNU/Linux machines [http://www.debian-administration.org/?article=70]). For the latter you have to introduce a network connectivity test script that will check if the kernel has started up the network subsystem properly and reboot the system if it did not[5]. This should prevent nasty surprises like updating the kernel and then realizing, after a reboot, that it did not detect or configure the network hardware properly and you need

[4] Unless you have installed a kernel metapackage like linux-image-2.6-686 which will always pull in the latest kernel minor revision for a kernel release and a given architecture.

[5] A sample script called testnet [http://www.debian-administration.org/articles/70/testnet] is available in the Remotely rebooting Debian GNU/Linux machines [http://www.debian-administration.org/?article=70] article. A more elaborate network connectivity testing script is available in this Testing network connectivity article. [http://www.debian-administration.org/?article=128]

to travel a long distance to bring the system up again. Of course, having the system serial console [6] in the system connected to a console or terminal server should also help debug reboot issues remotely.

Altere a BIOS (de novo)

Remember “Escolha uma senha para a BIOS”? Well, then you should now, once you do not need to boot from removable media, to change the default BIOS setup so that it *only* boots from the hard drive. Make sure you will not lose the BIOS password, otherwise, in the event of a hard disk failure you will not be able to return to the BIOS and change the setup so you can recover it using, for example, a CD-ROM.

Outro método mais conveniente, mas menos seguro, é alterar a configuração para ter o sistema inicializando a partir do disco rígido e, caso falhe, tentar a mídia removível. Por agora, isto é feito freqüentemente porque a maioria das pessoas não usam a senha de BIOS com freqüência; pois se esquecem dela facilmente.

Configurar a senha do LILO ou GRUB

Anybody can easily get a root-shell and change your passwords by entering

<name-of-your-bootimage> init=/bin/sh

at the boot prompt. After changing the passwords and rebooting the system, the person has unlimited root-access and can do anything he/she wants to the system. After this procedure you will not have root access to your system, as you do not know the root password.

Para se assegurar que isto não ocorra, você deverá definir uma senha para o gerenciador de partida. Escolha entre uma senha global ou uma senha para determinada imagem.

For LILO you need to edit the config file `/etc/lilo.conf` and add a **`password`** and **`restricted`** line as in the example below.

```
image=/boot/2.2.14-vmlinuz
   label=Linux
   read-only
   password=hackme
   restricted
```

Then, make sure that the configuration file is not world readable to prevent local users from reading the password. When done, rerun lilo. Omitting the `restricted` line causes lilo to always prompt for a password, regardless of whether LILO was passed parameters. The default permissions for `/etc/lilo.conf` grant read and write permissions to root, and enable read-only access for `lilo.conf`'s group, root.

If you use GRUB instead of LILO, edit `/boot/grub/menu.lst` and add the following two lines at the top (substituting, of course **`hackme`** with the desired password). This prevents users from editing the boot items. **`timeout 3`** specifies a 3 second delay before **grub** boots the default item.

```
timeout 3
password hackme
```

[6] Setting up a serial console is beyond the scope of this document, for more information read the Serial HOWTO [http://www.tldp.org/HOWTO/Serial-HOWTO.html] and the Remote Serial Console HOWTO [http://www.tldp.org/HOWTO/Remote-Serial-Console-HOWTO/index.html].

To further harden the integrity of the password, you may store the password in an encrypted form. The utility **grub-md5-crypt** generates a hashed password which is compatible with GRUB's encrypted password algorithm (MD5). To specify in **grub** that an MD5 format password will be used, use the following directive:

```
timeout 3
password --md5 $1$bw0ez$tljnxxKLfMzmnDVaQWgjP0
```

The --md5 parameter was added to instruct **grub** to perform the MD5 authentication process. The provided password is the MD5 encrypted version of hackme. Using the MD5 password method is preferable to choosing its clear-text counterpart. More information about **grub** passwords may be found in the grub-doc package.

Disable root prompt on the initramfs

Note: This applies to the default kernels provided for releases after Debian 3.1

Linux 2.6 kernels provide a way to access a root shell while booting which will be presented during loading the initramfs on error. This is helpful to permit the administrator to enter a rescue shell with root permissions. This shell can be used to manually load modules when autodetection fails. This behavior is the default for **initramfs-tools** generated initramfs. The following message will appear:

```
"ALERT!  /dev/sda1 does not exist.  Dropping to a shell!
```

In order to remove this behavior you need to set the following boot argument:*panic=0*. Add this to the variable `GRUB_CMDLINE_LINUX` in `/etc/default/grub` and issue **update-grub** or to the append section of `/etc/lilo.conf`.

Remover o aviso de root do kernel

Note: This does not apply to the kernels provided for Debian 3.1 as the timeout for the kernel delay has been changed to 0.

Linux 2.4 kernels provide a way to access a root shell while booting which will be presented just after loading the cramfs file system. A message will appear to permit the administrator to enter an executable shell with root permissions, this shell can be used to manually load modules when autodetection fails. This behavior is the default for **initrd**'s `linuxrc`. The following message will appear:

```
Press ENTER to obtain a shell (waits 5 seconds)
```

In order to remove this behavior you need to change `/etc/mkinitrd/mkinitrd.conf` and set:

```
# DELAY  The  number  of seconds the linuxrc script should wait to
# allow the user to interrupt it before the system is brought up
DELAY=0
```

Então gere novamente sua imagem do disco ram. Um exemplo de como fazer isto:

```
# cd /boot
```

```
# mkinitrd -o initrd.img-2.4.18-k7 /lib/modules/2.4.18-k7
```

ou (preferido):

```
# dpkg-reconfigure -plow kernel-image-2.4.x-yz
```

Restringindo o acesso de login no console

Some security policies might force administrators to log in to the system through the console with their user/password and then become superuser (with **su** or **sudo**). This policy is implemented in Debian by editing the `/etc/pam.d/login` and the `/etc/securetty` when using PAM:

`/etc/pam.d/login` In older Debian releases you would need to edit `login.defs`, and use the CONSOLE variable which defines a file or list of terminals on which root logins are allowed. enables the pam_securetty.so module. This module, when properly configured will not ask for a password when the root user tries to login on an insecure console, rejecting access as this user.

`securetty` The `/etc/securetty` is a configuration file that belongs to the login package. by adding/removing the terminals to which root access will be allowed. If you wish to allow only local console access then you need *console*, *ttyX* Or *ttyvX* in GNU/FreeBSD, and *ttyE0* in GNU/KNetBSD. and *vc/X* (if using *devfs* devices), you might want to add also *ttySX* Or *comX* in GNU/Hurd, *cuaaX* in GNU/FreeBSD, and *ttyXX* in GNU/KNetBSD. if you are using a serial console for local access (where X is an integer, you might want to have multiple instances. The default configuration for *Wheezy* The default configuration in *woody* includes 12 local tty and vc consoles, as well as the *console* device but does not allow remote logins. In *sarge* the default configuration provides 64 consoles for tty and vc consoles. includes many tty devices, serial ports, vc consoles as well as the X server and the *console* device. You can safely adjust this if you are not using that many consoles. You can confirm the virtual consoles and the tty devices you have by reviewing `/etc/inittab` Look for the *getty* calls. . For more information on terminal devices read the Text-Terminal-HOWTO [http://tldp.org/HOWTO/Text-Terminal-HOWTO-6.html]

When using PAM, other changes to the login process, which might include restrictions to users and groups at given times, can be configured in `/etc/pam.d/login`. An interesting feature that can be disabled is the possibility to login with null (blank) passwords. This feature can be limited by removing *nullok* from the line:

```
auth       required   pam_unix.so nullok
```

Restringindo reinicializações do sistema através da console

If your system has a keyboard attached to it anyone (yes *anyone*) with physical access to the system can reboot the system through it without login in just pressing the *Ctrl+Alt+Delete* keyboard combination, also known as the *three finger salute*. This might, or might not, adhere to your security policy.

This is aggravated in environments in which the operating system is running virtualised. In these environments, the possibility extends to users that have access to the virtual console (which might be accessed over the network). Also note that, in these environments, this keyboard combination is used constantly (to open a login shell in some GUI operating systems) and an administrator might *virtually* send it and force a system reboot.

There are two ways to restrict this:

- configure it so that only *allowed* users can reboot the system,
- disable this feature completely.

If you want to restrict this, you must check the `/etc/inittab` so that the line that includes **`ctrlaltdel`** calls **shutdown** with the **`-a`** switch.

The default in Debian includes this switch:

```
ca:12345:ctrlaltdel:/sbin/shutdown -t1 -a -r now
```

The **`-a`** switch, as the shutdown(8) manpage describes,makes it possible to allow *some* users to shutdown the system. For this the file `/etc/shutdown.allow` must be created and the administrator has to include there the name of users which can boot the system. When the *three finger salute* combination is pressed in a console the program will check if any of the users listed in the file are logged in. If none of them is, **shutdown** will *not* reboot the system.

If you want to disable the Ctrl+Alt+Del combination you just need to comment the line with the *ctrlaltdel* definition in the `/etc/inittab`.

Remember to run **`init q`** after making any changes to the `/etc/inittab` file for the changes to take effect.

Restricting the use of the Magic SysRq key

The *Magic SysRq key* is a key combination that allows users connected to the system console of a Linux kernel to perform some low-level commands. These low-level commands are sent by pressing simultaneously *Alt+SysRq* and a command key. The SysRq key in many keyboards is labeled as the *Print Screen* key.

Since the Etch release, the Magic SysRq key feature is enabled in the Linux kernel to allow console users certain privileges. You can confirm this by checking if the `/proc/sys/kernel/sysrq` exists and reviewing its value:

```
$ cat /proc/sys/kernel/sysrq
438
```

The default value shown above allows all of the SysRq functions except for the possibility of sending signals to processes. For example, it allow users connected to the console to remount all systems read-only, reboot the system or cause a kernel panic. In all the features are enabled, or in older kernels (earlier than 2.6.12) the value will be just 1.

You should disable this functionality ifaccess to the console is not restricted to authorised users: the console is connected to a modem line, there is easy physical access to the system or it is running in a virtualised environment and other users access the console. To do this edit the `/etc/sysctl.conf` and add the following lines:

```
# Disables the magic SysRq key
kernel.sysrq = 0
```

For more information, read security chapter in the Remote Serial Console HOWTO [http://tldp.org/HOWTO/Remote-Serial-Console-HOWTO/security-sysrq.html], Kernel SysRQ documentation [https://

www.kernel.org/doc/Documentation/admin-guide/sysrq.rst]. and the Magic_SysRq_key wikipedia entry [http://en.wikipedia.org/wiki/Magic_SysRq_key].

Montando partições do jeito certo

When mounting an `Ext` file system (`ext2`, `ext3` or `ext4`), there are several additional options you can apply to the mount call or to `/etc/fstab`. For instance, this is my fstab entry for the `/tmp` partition:

```
/dev/hda7    /tmp    ext2    defaults,nosuid,noexec,nodev    0    2
```

You see the difference in the options sections. The option `nosuid` ignores the setuid and setgid bits completely, while `noexec` forbids execution of any program on that mount point, and `nodev` ignores device files. This sounds great, but it:

- only applies to `ext2` or `ext3` file systems
- podem ser burlados facilmente

The `noexec` option prevents binaries from being executed directly, but was easily circumvented in earlier versions of the kernel:

```
alex@joker:/tmp# mount | grep tmp
/dev/hda7 on /tmp type ext2 (rw,noexec,nosuid,nodev)
alex@joker:/tmp# ./date
bash: ./date: Permission denied
alex@joker:/tmp# /lib/ld-linux.so.2 ./date
Sun Dec  3 17:49:23 CET 2000
```

Newer versions of the kernel do however handle the `noexec` flag properly:

```
angrist:/tmp# mount | grep /tmp
/dev/hda3 on /tmp type ext3 (rw,noexec,nosuid,nodev)
angrist:/tmp# ./date
bash: ./tmp: Permission denied
angrist:/tmp# /lib/ld-linux.so.2 ./date
./date: error while loading shared libraries: ./date: failed to map segment
from shared object: Operation not permitted
```

However, many script kiddies have exploits which try to create and execute files in `/tmp`. If they do not have a clue, they will fall into this pit. In other words, a user cannot be tricked into executing a trojanized binary in `/tmp` e.g. when `/tmp` is accidentally added into the local PATH.

Also be forewarned, some script might depend on `/tmp` being executable. Most notably, Debconf has (had?) some issues regarding this, for more information see http://bugs.debian.org/116448.

The following is a more thorough example. A note, though: `/var` could be set noexec, but some software [7] keeps its programs under in `/var`. The same applies to the nosuid option.

[7] Some of this includes the package manager dpkg since the installation (post,pre) and removal (post,pre) scripts are at `/var/lib/dpkg/` and Smartlist

```
/dev/sda6   /usr          ext3    defaults,ro,nodev       0       2
/dev/sda12  /usr/share    ext3    defaults,ro,nodev,nosuid        0       2
/dev/sda7   /var          ext3    defaults,nodev,usrquota,grpquota 0      2
/dev/sda8   /tmp          ext3    defaults,nodev,nosuid,noexec,usrquota,grpquota
/dev/sda9   /var/tmp      ext3    defaults,nodev,nosuid,noexec,usrquota,grpquota
/dev/sda10  /var/log      ext3    defaults,nodev,nosuid,noexec    0       2
/dev/sda11  /var/account  ext3    defaults,nodev,nosuid,noexec    0       2
/dev/sda13  /home         ext3    rw,nosuid,nodev,exec,auto,nouser,async,usrquota,
/dev/fd0    /mnt/fd0      ext3    defaults,users,nodev,nosuid,noexec      0
/dev/fd0    /mnt/floppy   vfat    defaults,users,nodev,nosuid,noexec      0
/dev/hda    /mnt/cdrom    iso9660 ro,users,nodev,nosuid,noexec            0
```

Setting `/tmp` noexec

Be careful if setting `/tmp` noexec when you want to install new software, since some programs might use it for installation. apt is one such program (see http://bugs.debian.org/116448) if not configured properly `APT::ExtractTemplates::TempDir` (see apt-extracttemplates(1)). You can set this variable in `/etc/apt/apt.conf` to another directory with exec privileges other than `/tmp`.

Definindo o /usr como somente-leitura

If you set `/usr` read-only you will not be able to install new packages on your Debian GNU/Linux system. You will have to first remount it read-write, install the packages and then remount it read-only. apt can be configured to run commands before and after installing packages, so you might want to configure it properly.

To do this modify `/etc/apt/apt.conf` and add:

```
DPkg
{
    Pre-Invoke  { "mount /usr -o remount,rw" };
    Post-Invoke { "mount /usr -o remount,ro" };
};
```

Note that the Post-Invoke may fail with a "/usr busy" error message. This happens mainly when you are using files during the update that got updated. You can find these programs by running

```
# lsof +L1
```

Stop or restart these programs and run the Post-Invoke manually. *Beware!* This means you'll likely need to restart your X session (if you're running one) every time you do a major upgrade of your system. You might want to reconsider whether a read-only `/usr` is suitable for your system. See also this discussion on debian-devel about read-only [http://lists.debian.org/debian-devel/2001/11/threads.html#00212].

Fornecendo acesso seguro ao usuário

Autenticação do Usuário: PAM

PAM (Pluggable Authentication Modules) allows system administrators to choose how applications authenticate users. Note that PAM can do nothing unless an application is compiled with support for PAM. Most of the applications that are shipped with Debian have this support built in (Debian did not have

PAM support before 2.2). The current default configuration for any PAM-enabled service is to emulate UNIX authentication (read `/usr/share/doc/libpam0g/Debian-PAM-MiniPolicy.gz` for more information on how PAM services *should* work in Debian).

Each application with PAM support provides a configuration file in `/etc/pam.d/` which can be used to modify its behavior:

- que método é usada para autenticação.
- que método é usada para sessões.
- como a checagem de senha se comportará.

The following description is far from complete, for more information you might want to read the Linux-PAM Guides [https://packages.debian.org/sid/libpam-doc] as a reference. This documentation is available in the system if you install the libpam-doc at `/usr/share/doc/libpam-doc/html/`.

PAM offers you the possibility to go through several authentication steps at once, without the user's knowledge. You could authenticate against a Berkeley database and against the normal `passwd` file, and the user only logs in if the authentication succeeds in both. You can restrict a lot with PAM, just as you can open your system doors very wide. So be careful. A typical configuration line has a control field as its second element. Generally it should be set to `requisite`, which returns a login failure if one module fails.

Password security in PAM

Review the `/etc/pam.d/common-password`, included by `/etc/pam.d/passwd`[8] This file is included by other files in `/etc/pam.d/` to define the behaviour of password use in subsystems that grant access to services in the machine, like the console login (login), graphical login managers (such as gdm or lightdm), and remote login (such as sshd). This definition is

You have to make sure that the pam_unix.so module uses the "sha512" option to use encrypted passwords. This is the default in Debian Squeeze.

The line with the definition of the pam_unix module will look something like:

```
password   [success=1 default=ignore]      pam_unix.so nullok obscure minlen=8 s
```

This definition:

- Enforces password encryption when storing passwords, using the SHA-512 hash function (option *sha512*),
- Enables password complexity checks (option *obscure*) as defined in the pam_unix(8) manpage,
- Imposes a minimum password length (option *min*) of 8.

You have to ensure that encrypted passwords are used in PAM applications, since this helps protect against dictionary cracks. Using encryption also makes it possible to use passwords longer than 8 characters.

Since this module is also used to define how passwords are changed (it is included by **chpasswd**) you can strengthen the password security in the system by installing libpam-cracklib and introducing this definition in the `/etc/pam.d/common-password` configuration file:

[8] In old Debian releases the configuration of the modules was defined directly in `/etc/pam.d/passwd`.

```
# Be sure to install libpam-cracklib first or you will not be able to log in
password   required     pam_cracklib.so retry=3 minlen=12 difok=3
password   [success=1 default=ignore]      pam_unix.so obscure minlen=8 sha512 u
```

So, what does this incantation do? The first line loads the cracklib PAM module, which provides password strength-checking, prompts for a new password with a minimum size [9] of 12 characters, and difference of at least 3 characters from the old password, and allows 3 retries. Cracklib depends on a wordlist package (such as wenglish, wspanish, wbritish, ...), so make sure you install one that is appropriate for your language or cracklib might not be useful to you at all.

The second line (using the pam_unix.so module) is the default configuration in Debian, as described above, save for the *use_authok* option. The *use_authok* option is required if pam_unix.so is stacked after pam_cracklib.so, and is used to hand over the password from the previous module. Otherwise, the user would be prompted for the password twice.

For more information about setting up Cracklib, read the pam_cracklib(8) manpage and the article Linux Password Security with pam_cracklib [http://www.deer-run.com/~hal/sysadmin/pam_cracklib.html] by Hal Pomeranz.

By enabling the cracklib PAM module you setup a policy that forces uses to use strong passwords.

Alternatively, you can setup and configure PAM modules to use double factor authentication such as: libpam-barada, libpam-google-authenticator, libpam-oath, libpam-otpw, libpam-poldi, libpam-usb or libpam-yubico. The configuration of these modules would make it possible to access the system using external authentication mechanisms such as smartcards, external USB keys, or One-Time-Passwords generated by external applications running, for example, in the user's mobile phone.

Please note that these restrictions apply to all users but *not* to the password changes done by the root user. The root user will be able to set up any password (any length or complexity) for personal use or others regardless of the restrictions defined here.

User access control in PAM

To make sure that the user root can only log into the system from local terminals, the following line should be enabled in `/etc/pam.d/login`:

```
auth     requisite  pam_securetty.so
```

Then you should modify the list of terminals on which direct root login is allowed in `/etc/securetty` (as described in “Restringindo o acesso de login no console”). Alternatively, you could enable the pam_access module and modify `/etc/security/access.conf` which allows for a more general and fine-tuned access control, but (unfortunately) lacks decent log messages (logging within PAM is not standardized and is particularly unrewarding problem to deal with). We'll return to `access.conf` a little later.

User limits in PAM

The following line should be enabled in `/etc/pam.d/login` to set up user resource limits.

[9] The minlen option is not entirely straightforward and is not exactly the number of characters in the password. A tradeoff can be defined between complexity and length by adjusting the "credit" parameters of different character classes. For more information read the pam_cracklib(8) manpage.

```
session  required   pam_limits.so
```

This restricts the system resources that users are allowed (see below in “Limiting resource usage: the `limits.conf` file”). For example, you could restrict the number of concurrent logins (of a given group of users, or system-wide), number of processes, memory size etc.

Control of su in PAM

If you want to protect **su**, so that only some people can use it to become root on your system, you need to add a new group "wheel" to your system (that is the cleanest way, since no file has such a group permission yet). Add root and the other users that should be able to **su** to the root user to this group. Then add the following line to `/etc/pam.d/su`:

```
auth        requisite   pam_wheel.so group=wheel debug
```

This makes sure that only people from the group "wheel" can use **su** to become root. Other users will not be able to become root. In fact they will get a denied message if they try to become root.

If you want only certain users to authenticate at a PAM service, this is quite easy to achieve by using files where the users who are allowed to login (or not) are stored. Imagine you only want to allow users 'ref' to log in via **ssh**. So you put them into `/etc/sshusers-allowed` and write the following into `/etc/pam.d/ssh`:

```
auth        required    pam_listfile.so item=user sense=allow file=/etc/sshusers
```

Temporary directories in PAM

Since there have been a number of so called insecure tempfile vulnerabilities, thttpd is one example (see DSA-883-1 [http://www.debian.org/security/2005/dsa-883]), the libpam-tmpdir is a good package to install. All you have to do is add the following to `/etc/pam.d/common-session`:

```
session    optional     pam_tmpdir.so
```

There has also been a discussion about adding this by default in Debian configuration, but it s. See http://lists.debian.org/debian-devel/2005/11/msg00297.html for more information.

Configuration for undefined PAM applications

Finally, but not least, create `/etc/pam.d/other` and enter the following lines:

```
auth     required       pam_securetty.so
auth     required       pam_unix_auth.so
auth     required       pam_warn.so
auth     required       pam_deny.so
account  required       pam_unix_acct.so
account  required       pam_warn.so
account  required       pam_deny.so
password required       pam_unix_passwd.so
password required       pam_warn.so
```

```
password required       pam_deny.so
session  required       pam_unix_session.so
session  required       pam_warn.so
session  required       pam_deny.so
```

Esta linhas lhe oferecerão uma boa configuração padrão para todas as aplicações que suportam PAM (o acesso é negado por padrão).

Limiting resource usage: the `limits.conf` file

You should really take a serious look into this file. Here you can define user resource limits. In old releases this configuration file was `/etc/limits.conf`, but in newer releases (with PAM) the `/etc/security/limits.conf` configuration file should be used instead.

If you do not restrict resource usage, *any* user with a valid shell in your system (or even an intruder who compromised the system through a service or a daemon going awry) can use up as much CPU, memory, stack, etc. as the system can provide. This *resource exhaustion* problem can be fixed by the use of PAM.

There is a way to add resource limits to some shells (for example, **bash** has **ulimit**, see bash(1)), but since not all of them provide the same limits and since the user can change shells (see chsh(1)) it is better to place the limits on the PAM modules as they will apply regardless of the shell used and will also apply to PAM modules that are not shell-oriented.

Resource limits are imposed by the kernel, but they need to be configured through the `limits.conf` and the PAM configuration of the different services need to load the appropriate PAM. You can check which services are enforcing limits by running:

```
$ find /etc/pam.d/ \! -name "*.dpkg*" | xargs -- grep limits |grep -v ":#"
```

Commonly, `login`, `ssh` and the graphic session managers (`gdm`, `kdm` or `xdm`) should enforce user limits but you might want to do this in other PAM configuration files, such as `cron`, to prevent system daemons from taking over all system resources.

The specific limits settings you might want to enforce depend on your system's resources, that's one of the main reasons why no limits are enforced in the default installation.

For example, the configuration example below enforces a 100 process limit for all users (to prevent *fork bombs*) as well as a limit of 10MB of memory per process and a limit of 10 simultaneous logins. Users in the `adm` group have higher limits and can produce core files if they want to (there is only a *soft* limit).

```
*               soft    core            0
*               hard    core            0
*               hard    rss             1000
*               hard    memlock         1000
*               hard    nproc           100
*               -       maxlogins       1
*               hard    data            102400
*               hard    fsize           2048
@adm            hard    core            100000
@adm            hard    rss             100000
@adm            soft    nproc           2000
@adm            hard    nproc           3000
@adm            hard    fsize           100000
```

```
@adm            -       maxlogins       10
```

These would be the limits a default user (including system daemons) would have:

```
$ ulimit -a
core file size        (blocks, -c) 0
data seg size         (kbytes, -d) 102400
file size             (blocks, -f) 2048
max locked memory     (kbytes, -l) 10000
max memory size       (kbytes, -m) 10000
open files                    (-n) 1024
pipe size          (512 bytes, -p) 8
stack size            (kbytes, -s) 8192
cpu time             (seconds, -t) unlimited
max user processes            (-u) 100
virtual memory        (kbytes, -v) unlimited
```

And these are the limits for an administrative user:

```
$ ulimit -a
core file size        (blocks, -c) 0
data seg size         (kbytes, -d) 102400
file size             (blocks, -f) 100000
max locked memory     (kbytes, -l) 100000
max memory size       (kbytes, -m) 100000
open files                    (-n) 1024
pipe size          (512 bytes, -p) 8
stack size            (kbytes, -s) 8192
cpu time             (seconds, -t) unlimited
max user processes            (-u) 2000
virtual memory        (kbytes, -v) unlimited
```

Mais detalhes podem ser lidos em:

- PAM reference guide for available modules [https://web.archive.org/web/20030601112932/http://www.kernel.org/pub/linux/libs/pam/Linux-PAM-html/pam-6.html]
- PAM configuration article [https://web.archive.org/web/20030217012148/http://www.samag.com/documents/s=1161/sam0009a/0009a.htm].
- Seifried's Securing Linux Step by Step [http://seifried.org/security/os/linux/20020324-securing-linux-step-by-step.html] on the *Limiting users overview* section.
- LASG [http://seifried.org/lasg/users/] in the *Limiting and monitoring users* section.

Ações de login do usuário: edite o `/etc/login.defs`

The next step is to edit the basic configuration and action upon user login. Note that this file is not part of the PAM configuration, it's a configuration file honored by login and **su** programs, so it doesn't make sense tuning it for cases where neither of the two programs are at least indirectly called (the **getty** program which sits on the consoles and offers the initial login prompt *does* invoke **login**).

```
  FAILLOG_ENAB        yes
```

Se ativar esta variável, as falhas nas tentativas de login serão registradas. É importante mantê-las para pegar alguém que tente fazer um ataque brute force.

```
LOG_UNKFAIL_ENAB     no
```

If you set this variable to 'yes' it will record unknown usernames if the login failed. It is best if you use 'no' (the default) since, otherwise, user passwords might be inadvertenly logged here (if a user mistypes and they enter their password as the username). If you set it to 'yes', make sure the logs have the proper permissions (640 for example, with an appropriate group setting such as adm).

```
SYSLOG_SU_ENAB       yes
```

This one enables logging of **su** attempts to `syslog`. Quite important on serious machines but note that this can create privacy issues as well.

```
SYSLOG_SG_ENAB       yes
```

The same as `SYSLOG_SU_ENAB` but applies to the **sg** program.

```
ENCRYPT_METHOD  SHA512
```

As stated above, encrypted passwords greatly reduce the problem of dictionary attacks, since you can use longer passwords. This definition has to be consistent with the value defined in `/etc/pam.d/common-password`.

User login actions: edit `/etc/pam.d/login`

You can adjust the login configuration file to implement an stricter policy. For example, you can change the default configuration and increase the delay time between login prompts. The default configuration sets a 3 seconds delay:

```
auth       optional   pam_faildelay.so  delay=3000000
```

Increasing the *delay* value to a higher value to make it harder to use the terminal to log in using brute force. If a wrong password is typed in, the possible attacker (or normal user!) has to wait longer seconds to get a new login prompt, which is quite time consuming when you test passwords. For example, if you set *delay=10000000*, users will have to wait 10 seconds if they type a wrong password.

In this file you can also set the system to present a message to users before a user logs in. The default is disabled, as shown below:

```
# auth       required   pam_issue.so issue=/etc/issue
```

If required by your security policy, this file can be used to show a standard message indicating that access to the system is restricted and user acess is logged. This kind of disclaimer might be required in some environments and jurisdictions. To enable it, just include the relevant information in the `/etc/issue`[10] file and uncomment the line enabling the pam_issue.so module in `/etc/pam.d/login`. In this file you can also enable additional features which might be relevant to apply local security policies such as:

[10] The default content of this file provides information about the operating system and version run by the system, which you might not want to provide to anonymous users.

- setting rules for which users can access at which times, by enabling the *pam_time.so* module and configuring `/etc/security/time.conf` accordingly (disabled by default),
- setup login sessions to use user limits as defined in `/etc/security/limits.conf` (enabled by default),
- present the user with the information of previous login information (enabled by default),
- print a message (`/etc/motd` and `/run/motd.dynamic`) to users after login in (enabled by default),

Restricting ftp: editing `/etc/ftpusers`

The `/etc/ftpusers` file contains a list of users who are not allowed to log into the host using ftp. Only use this file if you really want to allow ftp (which is not recommended in general, because it uses clear-text passwords). If your daemon supports PAM, you can also use that to allow and deny users for certain services.

FIXME (BUG): Is it a bug that the default `ftpusers` in Debian does *not* include all the administrative users (in base-passwd).

A convenient way to add all system accounts to the `/etc/ftpusers` is to run

```
$ awk -F : '{if ($3<1000) print $1}' /etc/passwd > /etc/ftpusers
```

Usando su

If you really need users to become the super user on your system, e.g. for installing packages or adding users, you can use the command **su** to change your identity. You should try to avoid any login as user root and instead use **su**. Actually, the best solution is to remove **su** and switch to the **sudo** mechanism which has a broader logic and more features than **su**. However, **su** is more common as it is used on many other Unices.

Usando o sudo

sudo allows the user to execute defined commands under another user's identity, even as root. If the user is added to `/etc/sudoers` and authenticates correctly, the commands defined in `/etc/sudoers` get enabled. Violations, such as incorrect passwords or trying to run a program you don't have permission for, are logged and mailed to root.

Desativação de acesso administrativo remoto

You should also modify `/etc/security/access.conf` to disallow remote logins to administrative accounts. This way users need to invoke **su** (or **sudo**) to use any administrative powers and the appropriate audit trace will always be generated.

You need to add the following line to `/etc/security/access.conf`, the default Debian configuration file has a sample line commented out:

```
-:wheel:ALL EXCEPT LOCAL
```

Remember to enable the pam_access module for every service (or default configuration) in `/etc/pam.d/` if you want your changes to `/etc/security/access.conf` honored.

Restringindo acessos de usuários

Algumas vezes você deve pensar que precisa ter seus usuários criados em seu sistema local para oferecer acesso a um determinado serviço (serviço de mensagens pop3 ou ftp). Antes de fazer isto, primeiro lembre-se que a implementação do PAM no Debian GNU/Linux lhe permite validar usuários em uma grande variedade de serviços de diretório externos (radius, ldap, etc.) através dos pacote libpam.

If users need to be created and the system can be accessed remotely take into account that users will be able to log in to the system. You can fix this by giving users a null (`/dev/null`) shell (it would need to be listed in `/etc/shells`). If you want to allow users to access the system but limit their movements, you can use the `/bin/rbash`, equivalent to adding the `-r` option in **bash** (*RESTRICTED SHELL* see bash(1)). Please note that even with restricted shell, a user that access an interactive program (that might allow execution of a subshell) could be able to bypass the limits of the shell.

Debian currently provides in the unstable release (and might be included in the next stable releases) the `pam_chroot` module (in the libpam-chroot). An alternative to it is to **chroot** the service that provides remote logging (**ssh**, **telnet**). [11]

If you wish to restrict *when* users can access the system you will have to customize `/etc/security/access.conf` for your needs.

Information on how to **chroot** users accessing the system through the **ssh** service is described in “Chroot environment for SSH”.

Auditoria do usuário

Se você é realmente paranóico, pode querer adicionar uma configuração em todo o sistema para auditar o que os usuários estão fazendo no sistema. Esta seção mostra algumas dicas de utilitários diversos que poderão ser usados.

Auditoria de entrada e saída com o script

You can use the **script** command to audit both what the users run and what are the results of those commands. You cannot setup **script** as a shell (even if you add it to `/etc/shells`). But you can have the shell initialization file run the following:

```
umask 077
exec script -q -a "/var/log/sessions/$USER"
```

Of course, if you do this system wide it means that the shell would not continue reading personal initialization files (since the shell gets overwritten by **script**). An alternative is to do this in the user's initialization files (but then the user could remove this, see the comments about this below)

You also need to setup the files in the audit directory (in the example `/var/log/sessions/`) so that users can write to it but cannot remove the file. This could be done, for example, by creating the user session files in advance and setting them with the *append-only* flag using **chattr**.

Uma alternativa útil para administradores de sistemas, que inclui informações sobre data, pode ser:

```
umask 077
exec script -q -a "/var/log/sessions/$USER-`date +%Y%m%d`"
```

[11] libpam-chroot has not been yet thoroughly tested, it does work for **login** but it might not be easy to set up the environment for other programs

Usando o arquivo de histórico do interpretador de comandos

If you want to review what does the user type in the shell (but not what the result of that is) you can setup a system-wide `/etc/profile` that configures the environment so that all commands are saved into a history file. The system-wide configuration needs to be setup in such a way that users cannot remove audit capabilities from their shell. This is somewhat shell specific so make sure that all users are using a shell that supports this.

For example, for bash, the `/etc/profile` could be set as follows [12] :

```
HISTFILE=~/.bash_history
HISTSIZE=10000
HISTFILESIZE=999999
# Don't let the users enter commands that are ignored
# in the history file
HISTIGNORE=""
HISTCONTROL=""
readonly HISTFILE
readonly HISTSIZE
readonly HISTFILESIZE
readonly HISTIGNORE
readonly HISTCONTROL
export HISTFILE HISTSIZE HISTFILESIZE HISTIGNORE HISTCONTROL
```

For this to work, the user can only append information to `.bash_history` file. You need *also* to set the *append-only* option using **chattr** program for `.bash_history` for all users. [13].

Note that you could introduce the configuration above in the user's `.profile`. But then you would need to setup permissions properly in such a way that prevents the user from modifying this file. This includes: having the user's home directories *not* belong to the user (since the user would be able to remove the file otherwise) but at the same time allow the user to read the `.profile` configuration file and write on the `.bash_history`. It would be good to set the *immutable* flag (also using **chattr**) for `.profile` too if you do it this way.

Auditoria completa do usuário com ferramentas de contabilização

The previous example is a simple way to configure user auditing but might be not useful for complex systems or for those in which users do not run shells at all (or exclusively). If this is your case, you need to look at acct, the accounting utilities. These utilities will log all the commands run by users or processes in the system, at the expense of disk space.

When activating accounting, all the information on processes and users is kept under `/var/account/`, more specifically in the `pacct`. The accounting package includes some tools (**sa**, **ac** and **lastcomm**) to analyse this data.

Outros métodos de auditoria do usuário

If you are completely paranoid and want to audit every user's command, you could take **bash** source code, edit it and have it send all that the user typed into another file. Or have ttysnoop constantly monitor any new ttys [14] and dump the output into a file. Other useful program is snoopy (see also github: https://github.com/

[12] Setting HISTSIZE to a very large number can cause issues under some shells since the history is kept in memory for every user session. You might be safer if you set this to a high-enough value and backup user's history files (if you need all of the user's history for some reason)
[13] Without the append-only flag users would be able to empty the contents of the history file running `> .bash_history`
[14] Ttys are spawned for local logins and remote logins through ssh and telnet

a2o/snoopy) which is a user-transparent program that hooks in as a library providing a wrapper around `execve()` calls, any command executed is logged to **syslogd** using the `authpriv` facility (usually stored at `/var/log/auth.log`).

Revisando perfis de usuários

If you want to *see* what users are actually doing when they logon to the system you can use the `wtmp` database that includes all login information. This file can be processed with several utilities, amongst them **sac** which can output a profile on each user showing in which timeframe they usually log on to the system.

No caso de ter a contabilização ativada, você também poderá usar as ferramentas fornecidas por ele para ser capaz de determinar quando os usuários acessam o sistema e o que eles executam.

Ajustando a umask dos usuários

Depending on your user policy you might want to change how information is shared between users, that is, what the default permissions of new files created by users are.

Debian's default `umask` setting is *022* this means that files (and directories) can be read and accessed by the user's group and by any other users in the system. This definition is set in the standard configuration file `/etc/profile` which is used by all shells.

If Debian's default value is too permissive for your system you will have to change the umask setting for all the shells. More restrictive umask settings include 027 (no access is allowed to new files for the *other* group, i.e. to other users in the system) or 077 (no access is allowed to new files to the members the user's group). Debian (by default[15]) creates one group per user so that only the user is included in its group. Consequently 027 and 077 are equivalent as the user's group contains only the user.

This change is set by defining a proper `umask` setting for all users. You can change this by introducing an **umask** call in the shell configuration files: `/etc/profile` (source by all Bourne-compatible shells), `/etc/csh.cshrc`, `/etc/csh.login`, `/etc/zshrc` and probably some others (depending on the shells you have installed on your system). You can also change the `UMASK` setting in `/etc/login.defs`, Of all of these the last one that gets loaded by the shell takes precedence. The order is: the default system configuration for the user's shell (i.e. `/etc/profile` and other system-wide configuration files) and then the user's shell (his `~/.profile`, `~/.bash_profile`, etc...). Some shells, however, can be executed with a *nologin* value which might skip sourcing some of those files. See your shell's manpage for additional information.

For connections that make use of **login** the UMASK definition in `/etc/login.defs` is used before any of the others. However, that value does not apply to user executed programs that do not use **login** such as those run through **su**, **cron** or **ssh**.

Don't forget to review and maybe modify the dotfiles under `/etc/skel/` since these will be new user's defaults when created with the **adduser** command. Debian default dotfiles do not include any **umask** call but if there is any in the dotfiles newly created users might a different value.

Note, however that users can modify their own `umask` setting if they want to, making it more permissive or more restricted, by changing their own dotfiles.

The libpam-umask package adjusts the users' default `umask` using PAM. Add the following, after installing the package, to `/etc/pam.d/common-session`:

[15] As defined in `/etc/adduser.conf` (USERGROUPS=yes). You can change this behaviour if you set this value to no, although it is not recommended

```
session    optional     pam_umask.so umask=077
```

Finally, you should consider changing root's default 022 umask (as defined in `/root/.bashrc`) to a more strict umask. That will prevent the system administrator from inadvertenly dropping sensitive files when working as root to world-readable directories (such as `/tmp`) and having them available for your average user.

Limitando o que os usuários podem ver/acessar

FIXME: Content needed. Describe the consequences of changing packages permissions when upgrading (an admin this paranoid should **chroot** his users BTW) if not using **dpkg-statoverride**.

If you need to grant users access to the system with a shell think about it very carefully. A user can, by default unless in a severely restricted environment (like a `chroot` jail), retrieve quite a lot of information from your system including:

- some configuration files in `/etc`. However, Debian's default permissions for some sensitive files (which might, for example, contain passwords), will prevent access to critical information. To see which files are only accessible by the root user for example

  `find /etc -type f -a -perm 600 -a -uid 0`

  as superuser.

- your installed packages, either by looking at the package database, at the `/usr/share/doc` directory or by guessing by looking at the binaries and libraries installed in your system.

- some log files at `/var/log`. Note also that some log files are only accessible to root and the `adm` group (try

  `find /var/log -type f -a -perm 640`

) and some are even only available to the root user (try

  ```
  find /var/log -type f -a -perm
      600 -a -uid 0
  ```

).

O que um usuário pode ver em seu sistema? Provavelmente muitas coisas, tente isto (faça uma breve parada):

```
find / -type f -a -perm +006 2>/dev/null
find / -type d -a -perm +007 2>/dev/null
```

The output is the list of files that a user can *see* and the accessable directories.

Limitando acesso a outras informações de usuários

Se você ainda permite acesso a shell para os usuários você deverá querer limitar que informações eles podem ver de outros usuários. Os usuários com acesso a shell têm a tendência de criar um número de arquivos dentro do seu diretório pessoal: caixas de correio, documentos pessoais, configurações de aplicativos do X/GNOME/KDE...

In Debian each user is created with one associated group, and no two users belong to the same group. This is the default behavior: when an user account is created, a group of the same name is created too, and the

user is assigned to it. This avoids the concept of a common *users* group which might make it more difficult for users to hide information from other users.

However, users' `$HOME` directories are created with 0755 permissions (group-readable and world-readable). The group permissions is not an issue since only the user belongs to the group, however the world permissions might (or might not) be an issue depending on your local policy.

You can change this behavior so that user creation provides different `$HOME` permissions. To change the behavior for *new* users when they get created, change *DIR_MODE* in the configuration file `/etc/adduser.conf` to 0750 (no world-readable access).

Users can still share information, but not directly in their `$HOME` directories unless they change its permissions.

Note that disabling world-readable home directories will prevent users from creating their personal web pages in the `~/public_html` directory, since the web server will not be able to read one component in the path - namely their `$HOME` directory. If you want to permit users to publish HTML pages in their `~/public_html`, then change *DIR_MODE* to 0751. This will allow the web server to access the final `public_html` directory (which itself should have a mode of 0755) and provide the content published by users. Of course, we are only talking about a default configuration here; users can generally tune modes of their own files completely to their liking, or you could keep content intended for the web in a separate location which is not a subdirectory of user's `$HOME` directory.

Gerando senhas de usuários

There are many cases when an administrator needs to create many user accounts and provide passwords for all of them. Of course, the administrator could easily just set the password to be the same as the user's account name, but that would not be very sensitive security-wise. A better approach is to use a password generating program. Debian provides makepasswd, apg and pwgen packages which provide programs (the name is the same as the package) that can be used for this purpose. **Makepasswd** will generate true random passwords with an emphasis on security over pronounceability while **pwgen** will try to make meaningless but pronounceable passwords (of course this might depend on your mother language). **Apg** has algorithms to provide for both (there is a client/server version for this program but it is not included in the Debian package).

Passwd does not allow non-interactive assignation of passwords (since it uses direct tty access). If you want to change passwords when creating a large number of users you can create them using **adduser** with the `--disabled-login` option and then use **usermod** or **chpasswd**[16] (both from the passwd package so you already have them installed). If you want to use a file with all the information to make users as a batch process you might be better off using **newusers**.

Verificando senhas de usuários

User passwords can sometimes become the *weakest link* in the security of a given system. This is due to some users choosing weak passwords for their accounts (and the more of them that have access to it the greater the chances of this happening). Even if you established checks with the cracklib PAM module and password limits as described in “Autenticação do Usuário: PAM” users will still be able to use weak passwords. Since user access might include remote shell access (over **ssh**, hopefully) it's important to make password guessing as hard as possible for the remote attackers, especially if they were somehow able to collect important information such as usernames or even the `passwd` and `shadow` files themselves.

[16] **Chpasswd** cannot handle MD5 password generation so it needs to be given the password in encrypted form before using it, with the

`-e`

option.

A system administrator must, given a big number of users, check if the passwords they have are consistent with the local security policy. How to check? Try to crack them as an attacker would if having access to the hashed passwords (the `/etc/shadow` file).

An administrator can use john or crack (both are brute force password crackers) together with an appropriate wordlist to check users' passwords and take appropriate action when a weak password is detected. You can search for Debian GNU packages that contain word lists using **apt-cache search wordlist**, or visit some Internet wordlist sites.

Logout de usuários ociosos

Usuários inativos geralmente são um risco de segurança, um usuário pode estar inativo porque saiu para comer ou porque ocorreu um problema com sua conexão remota, que não foi restabelecida. Por alguma razão, os usuários inativos podem levar a um comprometimento do sistema:

- porque o console do usuário pode ser destravado e pode ser acessado por um intruso.
- because an attacker might be able to re-attach to a closed network connection and send commands to the remote shell (this is fairly easy if the remote shell is not encrypted as in the case of **telnet**).

Some remote systems have even been compromised through an idle (and detached) **screen**.

A desconexão automática de usuários idle é geralmente parte da política local de segurança que deve ser forçada. Existem várias formas de se fazer isto:

- If bash is the user shell, a system administrator can set a default `TMOUT` value (see bash(1)) which will make the shell automatically log off remote idle users. Note that it must be set with the `-o` option or users will be able to change (or unset) it.
- Install timeoutd and configure `/etc/timeouts` according to your local security policy. The daemon will watch for idle users and time out their shells accordingly.
- Install autolog and configure it to remove idle users.

The **timeoutd** or **autolog** daemons are the preferred method since, after all, users can change their default shell or can, after running their default shell, switch to another (uncontrolled) shell.

Usando os tcpwrappers

TCP wrappers were developed when there were no real packet filters available and access control was needed. Nevertheless, they're still very interesting and useful. The TCP wrappers allow you to allow or deny a service for a host or a domain and define a default allow or deny rule (all performed on the application level). If you want more information take a look at hosts_access(5) manual page.

Muitos dos serviços instalados no Debian são executados de duas formas:

- launched through the tcpwrapper service (`tcpd`)
- compilados com o suporte a libwrapper embutido

On the one hand, for services configured in `/etc/inetd.conf` (this includes **telnet**, **ftp**, **netbios**, **swat** and **finger**) you will see that the configuration file executes **/usr/sbin/tcpd** first. On the other hand, even if a service is not launched by the **inetd** superdaemon, support for the tcp wrappers rules can be compiled into it. Services compiled with tcp wrappers in Debian include **ssh**, **portmap**, **in.talk**, **rpc.statd**, **rpc.mountd**, **gdm**, **oaf** (the GNOME activator daemon), **nessus** and many others.

To see which packages use tcpwrappers [17] try:

```
$ apt-cache rdepends libwrap0
```

Take this into account when running **tcpdchk** (a very useful TCP wrappers config file rule and syntax checker). When you add stand-alone services (that are directly linked with the wrapper library) into the `hosts.deny` and `hosts.allow` files, **tcpdchk** will warn you that it is not able to find the mentioned services since it only looks for them in `/etc/inetd.conf` (the manpage is not totally accurate here).

Now, here comes a small trick, and probably the smallest intrusion detection system available. In general, you should have a decent firewall policy as a first line, and tcp wrappers as the second line of defense. One little trick is to set up a `SPAWN` [18] command in `/etc/hosts.deny` that sends mail to root whenever a denied service triggers wrappers:

```
ALL: ALL: SPAWN ( \
  echo -e "\n\
  TCP Wrappers\: Connection refused\n\
  By\: $(uname -n)\n\
  Process\: %d (pid %p)\n\
  User\: %u\n\
  Host\: %c\n\
  Date\: $(date)\n\
" | /usr/bin/mail -s "Connection to %d blocked" root) &
```

Beware: The above printed example is open to a DoS attack by making many connections in a short period of time. Many emails mean a lot of file I/O by sending only a few packets.

A importância dos logs e alertas

É fácil ver que o tratamento de mensagens de logs e alertas é um assunto importante em um sistema seguro. Suponha que um sistema está perfeitamente configurado e é 99% seguro. Se a probabilidade de 1% do ataque ocorrer e não existir medidas de segurança no lugar, para primeiro detectar e segundo disparar alarmes, o sistema não estará bem seguro.

Debian GNU/Linux provides some tools to perform log analysis, most notably swatch, [19] logcheck or log-analysis (all will need some customisation to remove unnecessary things from the report). It might also be useful, if the system is nearby, to have the system logs printed on a virtual console. This is useful since you can (from a distance) see if the system is behaving properly. Debian's `/etc/syslog.conf` comes with a commented default configuration; to enable it uncomment the lines and restart **syslogd** (**`/etc/init.d/syslogd restart`**):

```
daemon,mail.*;\
      news.=crit;news.=err;news.=notice;\
      *.=debug;*.=info;\
      *.=notice;*.=warn       /dev/tty8
```

[17] On older Debian releases you might need to do this:

```
$ apt-cache showpkg libwrap0 | egrep '^[[:space:]]' | sort -u | \
      sed 's/,libwrap0$//;s/^[[:space:]]\+//'
```

[18] be sure to use uppercase here since *spawn* will not work

[19] there's a very good article on it written by http://www.spitzner.net/swatch.html

To colorize the logs, you could take a look at colorize, ccze or glark. There is a lot to log analysis that cannot be fully covered here, so a good information resource would be books should as http://books.google.com/books?id=UyktqN6GnWEC. In any case, even automated tools are no match for the best analysis tool: your brain.

Using and customizing logcheck

The **logcheck** package in Debian is divided into the three packages logcheck (the main program), logcheck-database (a database of regular expressions for the program) and logtail (prints loglines that have not yet been read). The Debian default (in `/etc/cron.d/logcheck`) is that **logcheck** is run every hour and after reboots.

This tool can be quite useful if properly customized to alert the administrator of unusual system events. **Logcheck** can be fully customized so that it sends mails based on events found in the logs and worthy of attention. The default installation includes profiles for ignored events and policy violations for three different setups (workstation, server and paranoid). The Debian package includes a configuration file `/etc/logcheck/logcheck.conf`, sourced by the program, that defines which user the checks are sent to. It also provides a way for packages that provide services to implement new policies in the directories: `/etc/logcheck/cracking.d/_packagename_`, `/etc/logcheck/violations.d/_packagename_`, `/etc/logcheck/violations.ignore.d/_packagename_`, `/etc/logcheck/ignore.d.paranoid/_packagename_`, `/etc/logcheck/ignore.d.server/_packagename_`, and `/etc/logcheck/ignore.d.workstation/_packagename_`. However, not many packages currently do so. If you have a policy that can be useful for other users, please send it as a bug report for the appropriate package (as a *wishlist* bug). For more information read `/usr/share/doc/logcheck/README.Debian`.

The best way to configure **logcheck** is to edit its main configuration file `/etc/logcheck/logcheck.conf` after installation. Change the default user (root) to whom reports should be mailed. You should set the reportlevel in there, too. logcheck-database has three report levels of increasing verbosity: workstation, server, paranoid. "server" being the default level, paranoid is only recommended for high-security machines running as few services as possible and workstation for relatively sheltered, non-critical machines. If you wish to add new log files just add them to `/etc/logcheck/logcheck.logfiles`. It is tuned for default syslog install.

Once this is done you might want to check the mails that are sent, for the first few days/weeks/months. If you find you are sent messages you do not wish to receive, just add the regular expressions (see regex(7) and egrep(1)) that correspond to these messages to the `/etc/logcheck/ignore.d.reportlevel/local`. Try to match the whole logline. Details on howto write rules are explained in `/usr/share/doc/logcheck-database/README.logcheck-database.gz`. It's an ongoing tuning process; once the messages that are sent are always relevant you can consider the tuning finished. Note that if **logcheck** does not find anything relevant in your system it will not mail you even if it does run (so you might get a mail only once a week, if you are lucky).

Configurando para onde os alertas são enviados

Debian comes with a standard syslog configuration (in `/etc/syslog.conf`) that logs messages to the appropriate files depending on the system facility. You should be familiar with this; have a look at the `syslog.conf` file and the documentation if not. If you intend to maintain a secure system you should be aware of where log messages are sent so they do not go unnoticed.

Por exemplo, o envio de mensagens para o console também é uma configuração interessante para muitos sistemas a nível de produção. Mas para muitos do sistemas também é importante adicionar uma nova máquina que servirá de servidor de logs (i.e. ela receberá os logs de todos os outros sistemas).

Root's mail should be considered also, many security controls (like snort) send alerts to root's mailbox. This mailbox usually points to the first user created in the system (check `/etc/aliases`). Take care to send root's mail to some place where it will be read (either locally or remotely).

Existem outras contas e aliases em seu sistema. Em um sistema pequeno, é provavelmente o método mais simples de ter certeza que todos estes aliases apontam para a senha de root, e aquele e-mail do root é redirecionado para a caixa de mensagens pessoal do administrador do sistema.

FIXME: It would be interesting to tell how a Debian system can send/receive SNMP traps related to security problems (jfs). Check: snmptrapfmt, snmp and snmpd.

Usando um servidor de logs

A loghost is a host which collects syslog data remotely over the network. If one of your machines is cracked, the intruder is not able to cover the tracks, unless hacking the loghost as well. So, the loghost should be especially secure. Making a machine a loghost is simple. Just start the **syslogd** with

```
syslogd -r
```

and a new loghost is born. In order to do this permanently in Debian, edit `/etc/default/syslogd` and change the line

```
SYSLOGD=""
```

to

```
SYSLOGD="-r"
```

Next, configure the other machines to send data to the loghost. Add an entry like the following to `/etc/syslog.conf`:

```
  facility.level              @your_loghost
```

See the documentation for what to use in place of *facility* and *level* (they should not be entered verbatim like this). If you want to log everything remotely, just write:

```
  *.*                         @your_loghost
```

into your `syslog.conf`. Logging remotely as well as locally is the best solution (the attacker might presume to have covered his tracks after deleting the local log files). See the syslog(3), syslogd(8) and syslog.conf(5) manpages for additional information.

Permissões dos arquivos de log

It is not only important to decide how alerts are used, but also who has read/modify access to the log files (if not using a remote loghost). Security alerts which the attacker can change or disable are not worth much in the event of an intrusion. Also, you have to take into account that log files might reveal quite a lot of information about your system to an intruder who has access to them.

Some log file permissions are not perfect after the installation (but of course this really depends on your local security policy). First `/var/log/lastlog` and `/var/log/faillog` do not need to be readable by normal users. In the `lastlog` file you can see who logged in recently, and in the `faillog` you see

a summary of failed logins. The author recommends **chmod 660** for both. Take a brief look at your log files and decide very carefully which log files to make readable/writable for a user with a UID other than 0 and a group other than 'adm' or 'root'. You can easily check this in your system with:

```
#  find /var/log -type f -exec ls -l {} \; | cut -c 17-35 |sort -u
(see to what users do files in /var/log belong)
#  find /var/log -type f -exec ls -l {} \; | cut -c 26-34 |sort -u
(see to what groups do files in /var/log belong)
# find /var/log -perm +004
(files which are readable by any user)
#  find /var/log \! -group root \! -group adm -exec ls -ld {} \;
(files which belong to groups not root or adm)
```

Para personalizar a forma que os arquivos de log são criados, você provavelmente terá que personalizar o programa que os gera. Se os arquivos de log forem rotacionados, no entanto, você poderá personalizar o comportamento do rotacionamento e da criação.

Adicionando patches no kernel

O Debian GNU/Linux oferece alguns dos patches para o kernel do Linux que aumentam sua segurança. Estes incluem:

- Linux Intrusion Detection [http://www.lids.org] provided in the kernel-patch-2.4-lids package. This kernel patch makes the process of hardening your Linux system easier by allowing you to restrict, hide and protect processes, even from root. It implements mandatory access control capabilities.

- Linux Trustees [http://trustees.sourceforge.net/], provided in package trustees. This patch adds a decent advanced permissions management system to your Linux kernel. Special objects (called trustees) are bound to every file or directory, and are stored in kernel memory, which allows fast lookup of all permissions.

- NSA Enhanced Linux (in package selinux). Backports of the SElinux-enabled packages are available at https://salsa.debian.org/selinux-team. More information available at SElinux in Debian Wiki page [http://wiki.debian.org/SELinux], at Manoj Srivastava's [http://www.golden-gryphon.com/software/security/selinux.xhtml] and Russell Cookers's [http://www.coker.com.au/selinux/] SElinux websites.

- The kernel patch http://people.redhat.com/mingo/exec-shield provided in the kernel-patch-exec-shield package. This patch provides protection against some buffer overflows (stack smashing attacks).

- The Grsecurity patch [http://www.grsecurity.net/], provided by the kernel-patch-2.4-grsecurity and kernel-patch-grsecurity2 packages [20] implements Mandatory Access Control through RBAC, provides buffer overflow protection through PaX, ACLs, network randomness (to make OS fingerprinting more difficult) and many more features [http://www.grsecurity.net/features.php].

[20] Notice that this patch conflicts with patches already included in Debian's 2.4 kernel source package. You will need to use the stock vanilla kernel. You can do this with the following steps:

```
# apt-get install kernel-source-2.4.22 kernel-patch-debian-2.4.22
# tar xjf /usr/src/kernel-source-2.4.22.tar.bz2
# cd kernel-source-2.4.22
# /usr/src/kernel-patches/all/2.4.22/unpatch/debian
```

For more information see http://bugs.debian.org/194225, http://bugs.debian.org/199519, http://bugs.debian.org/206458, http://bugs.debian.org/203759, http://bugs.debian.org/204424, http://bugs.debian.org/210762, http://bugs.debian.org/211213, and the http://lists.debian.org/debian-devel/2003/09/msg01133.html

- The kernel-patch-adamantix provides the patches developed for Adamantix [http://www.adamantix.org/], a Debian-based distribution. This kernel patch for the 2.4.x kernel releases introduces some security features such as a non-executable stack through the use of http://pageexec.virtualave.net/ and mandatory access control based on http://www.rsbac.org/. Other features include: http://www.vanheusden.com/ Linux/sp/, AES encrypted loop device, MPPE support and an IPSEC v2.6 backport.

- cryptoloop-source. This patches allows you to use the functions of the kernel crypto API to create encrypted filesystems using the loopback device.

- IPSEC kernel support (in package linux-patch-openswan). If you want to use the IPsec protocol with Linux, you need this patch. You can create VPNs with this quite easily, even to Windows machines, as IPsec is a common standard. IPsec capabilities have been added to the 2.5 development kernel, so this feature will be present by default in the future Linux Kernel 2.6. Homepage: http://www.openswan.org. *FIXME*: The latest 2.4 kernels provided in Debian include a backport of the IPSEC code from 2.5. Comment on this.

The following security kernel patches are only available for old kernel versions in woody and are deprecated:

- http://acl.bestbits.at/ (ACLs) for Linux provided in the package kernel-patch-acl. This kernel patch adds access control lists, an advanced method for restricting access to files. It allows you to control fine-grain access to files and directory.

- The http://www.openwall.com/linux/ linux kernel patch by Solar Designer, provided in the kernel-patch-2.2.18-openwall package. This is a useful set of kernel restrictions, like restricted links, FIFOs in `/tmp`, a restricted `/proc` file system, special file descriptor handling, non-executable user stack area and other features. Note: This package applies to the 2.2 release, no packages are available for the 2.4 release patches provided by Solar.

- kernel-patch-int. This patch also adds cryptographic capabilities to the Linux kernel, and was useful with Debian releases up to Potato. It doesn't work with Woody, and if you are using Sarge or a newer version, you should use a more recent kernel which includes these features already.

However, some patches have not been provided in Debian yet. If you feel that some of these should be included please ask for it at the http://www.debian.org/devel/wnpp/.

Protegendo-se contra estouros de buffer

Buffer overflow is the name of a common attack to software [21] which makes use of insufficient boundary checking (a programming error, most commonly in the C language) in order to execute machine code through program inputs. These attacks, against server software which listen to connections remotely and against local software which grant higher privileges to users (`setuid` or `setgid`) can result in the compromise of any given system.

Existem basicamente quatro métodos de se proteger contra estouro de buffer:

- patch the kernel to prevent stack execution. You can use either: Exec-shield, OpenWall or PaX (included in the Grsecurity and Adamantix patches).

- corrigir o código fonte usando ferramentas para encontrar fragmentos de onde pode introduzir esta vulnerabilidade.

[21] So common, in fact, that they have been the basis of 20% of the reported security vulnerabilities every year, as determined by http://icat.nist.gov/icat.cfm?function=statistics

- recompile the source code to introduce proper checks that prevent overflows, using the http://www.research.ibm.com/trl/projects/security/ssp/ patch for GCC (which is used by http://www.adamantix.org)

Debian GNU/Linux, as of the 3.0 release, provides software to introduce all of these methods except for the protection on source code compilation (but this has been requested in http://bugs.debian.org/213994).

Notice that even if Debian provided a compiler which featured stack/buffer overflow protection all packages would need to be recompiled in order to introduce this feature. This is, in fact, what the Adamantix distribution does (among other features). The effect of this new feature on the stability of software is yet to be determined (some programs or some processor architectures might break due to it).

In any case, be aware that even these workarounds might not prevent buffer overflows since there are ways to circumvent these, as described in phrack's magazine http://packetstorm.linuxsecurity.com/mag/phrack/phrack58.tar.gz or in CORE's Advisory http://online.securityfocus.com/archive/1/269246.

If you want to test out your buffer overflow protection once you have implemented it (regardless of the method) you might want to install the paxtest and run the tests it provides.

Patches de kernel para proteção contra estouros de buffer

Kernel patches related to buffer overflows include the Openwall patch provides protection against buffer overflows in 2.2 linux kernels. For 2.4 or newer kernels, you need to use the Exec-shield implementation, or the PaX implementation (provided in the grsecurity patch, kernel-patch-2.4-grsecurity, and in the Adamantix patch, kernel-patch-adamantix). For more information on using these patches read the the section “Adicionando patches no kernel”.

Testando problemas de estouro em programas

The use of tools to detect buffer overflows requires, in any case, of programming experience in order to fix (and recompile) the code. Debian provides, for example: bfbtester (a buffer overflow tester that brute-forces binaries through command line and environment overflows). Other packages of interest would also be rats, pscan, flawfinder and splint.

Transferência segura de arquivos

During normal system administration one usually needs to transfer files in and out from the installed system. Copying files in a secure manner from a host to another can be achieved by using the ssh server package. Another possibility is the use of ftpd-ssl, a ftp server which uses the *Secure Socket Layer* to encrypt the transmissions.

Any of these methods need special clients. Debian does provide client software, such as **scp** from the ssh package, which works like **rcp** but is encrypted completely, so the *bad guys* cannot even find out WHAT you copy. There is also a ftp-ssl package for the equivalent server. You can find clients for these software even for other operating systems (non-UNIX), **putty** and **winscp** provide secure copy implementations for any version of Microsoft's operating system.

Note that using **scp** provides access to the users to all the file system unless **chroot**'ed as described in “Executando o ssh em uma jaula chroot”. FTP access can be **chroot**'ed, probably easier depending on you chosen daemon, as described in “Tornando o FTP mais seguro”. If you are worried about users browsing your local files and want to have encrypted communication you can either use an ftp daemon with SSL support or combine clear-text ftp and a VPN setup (see “Redes Privadas Virtuais (VPN)”).

File system limits and control

Usando quotas

É importante se ter uma boa política de quotas, pois ela evita que os usuários ocupem todo o(s) disco(s) rígido(s).

Você poderá usar dois sistemas diferentes de quota: quota do usuário e quota do grupo. Você provavelmente notará que limites de quota de usuários definem o espaço que o usuário pode utilizar, a quota de grupo é equivalente para grupos. Mantenha isto em mente quando estiver trabalhando com tamanhos de quota.

Existem alguns pontos importantes que devem ser pensados sobre a configuração de um sistema de quotas:

- Mantenha as quotas suficientemente pequenas, assim os usuários não poderão acabar com todo seu espaço em disco.
- Mantenha as quotas grande o bastante, assim os usuarios não se importarão ou sua quota de e-mails os proibirá de receber mensagens por um longo período.
- Use quotas on all user-writable areas, on `/home` as well as on `/tmp`.

Cada partição ou diretório no qual os usuários tem acesso completo a gravação deverão ter a quota ativada. Calcule e defina um tamanho de quota funcional para estas partições e diretórios que combinam utilização e segurança.

So, now you want to use quotas. First of all you need to check whether you enabled quota support in your kernel. If not, you will need to recompile it. After this, control whether the package quota is installed. If not you will need this one as well.

Enabling quota for the respective file systems is as easy as modifying the `defaults` setting to `defaults,usrquota` in your `/etc/fstab` file. If you need group quota, substitute `usrquota` to `grpquota`. You can also use them both. Then create empty quota.user and quota.group files in the roots of the file systems you want to use quotas on (e.g.

```
touch
/home/quota.user /home/quota.group
```

for a `/home` file system).

Restart **quota** by doing

```
/etc/init.d/quota stop;/etc/init.d/quota
        start
```

. Now quota should be running, and quota sizes can be set.

Editing quotas for a specific user can be done by

```
edquota -u <user>
```

. Group quotas can be modified with

```
edquota -g <group>
```

. Then set the soft and hard quota and/or inode quotas as needed.

For more information about quotas, read the quota man page, and the quota mini-howto (`/usr/share/doc/HOWTO/en-html/mini/Quota.html`). You may also want to look at `pam_limits.so`.

The ext2 filesystem specific attributes (chattr/lsattr)

In addition to the usual Unix permissions, the ext2 and ext3 filesystems offer a set of specific attributes that give you more control over the files on your system. Unlike the basic permissions, these attributes are not displayed by the usual **ls -l** command or changed using **chmod**, and you need two other utilities, **lsattr** and **chattr** (in package e2fsprogs) to manage them. Note that this means that these attributes will usually not be saved when you backup your system, so if you change any of them, it may be worth saving the successive **chattr** commands in a script so that you can set them again later if you have to restore a backup.

Entre todos os atributos disponíveis, os dois abaixo são os mais importantes para aumentar a segurança e são referenciados pelas letras 'i' e 'a' e podem ser somente definidos (ou removidos) pelo superusuário:

- O atributo 'i' ('imutável'): um arquivo com este atributo não pode ser modificado, excluído ou renomeado, e nenhum link poderá ser criado para ele, até mesmo pelo superusuário.

- The 'a' attribute ('append'): this attribute has the same effect that the immutable attribute, except that you can still open the file in append mode. This means that you can still add more content to it but it is impossible to modify previous content. This attribute is especially useful for the log files stored in `/var/log/`, though you should consider that they get moved sometimes due to the log rotation scripts.

These attributes can also be set for directories, in which case everyone is denied the right to modify the contents of a directory list (e.g. rename or remove a file, ...). When applied to a directory, the append attribute only allows file creation.

It is easy to see how the 'a' attribute improves security, by giving to programs that are not running as the superuser the ability to add data to a file without modifying its previous content. On the other hand, the 'i' attribute seems less interesting: after all, the superuser can already use the basic Unix permissions to restrict access to a file, and an intruder that would get access to the superuser account could always use the **chattr** program to remove the attribute. Such an intruder may first be confused when noticing not being able to remove a file, but you should not assume blindness - after all, the intruder got into your system! Some manuals (including a previous version of this document) suggest to simply remove the **chattr** and **lsattr** programs from the system to increase security, but this kind of strategy, also known as "security by obscurity", is to be absolutely avoided, since it provides a false sense of security.

A secure way to solve this problem is to use the capabilities of the Linux kernel, as described in “Defesa pró-ativa”. The capability of interest here is called `CAP_LINUX_IMMUTABLE`: if you remove it from the capabilities bounding set (using for example the command **lcap `CAP_LINUX_IMMUTABLE`**) it won't be possible to change any 'a' or 'i' attribute on your system anymore, even by the superuser ! A complete strategy could be as follows:

- Defina os atributos 'a' e 'i' nos arquivos que deseja;

- Add the command **lcap `CAP_LINUX_IMMUTABLE`** (as well as **lcap `CAP_SYS_MODULE`**, as suggested in “Defesa pró-ativa”) to one of the startup scripts;

- Set the 'i' attribute on this script and other startup files, as well as on the **lcap** binary itself;

- Execute manualmente o comando acima (ou reinicie o seu sistema para ter certeza que tudo funciona como planejado).

Now that the capability has been removed from the system, an intruder cannot change any attribute on the protected files, and thus cannot change or remove the files. If the machine is forced to reboot (which is the only way to restore the capabilities bounding set), it will easily be detected, and the capability will be removed again as soon as the system restarts anyway. The only way to change a protected file would be to boot the system in single-user mode or using another bootdisk, two operations that require physical access to the machine !

Verificando a integridade do sistema de arquivos

Are you sure `/bin/login` on your hard drive is still the binary you installed there some months ago? What if it is a hacked version, which stores the entered password in a hidden file or mails it in clear-text version all over the Internet?

The only method to have some kind of protection is to check your files every hour/day/month (I prefer daily) by comparing the actual and the old md5sum of this file. Two files cannot have the same md5sum (the MD5 digest is 128 bits, so the chance that two different files will have the same md5sum is roughly one in 3.4e3803), so you're on the safe site here, unless someone has also hacked the algorithm that creates md5sums on that machine. This is, well, extremely difficult and very unlikely. You really should consider this auditing of your binaries as very important, since it is an easy way to recognize changes at your binaries.

Common tools used for this are sxid, aide (Advanced Intrusion Detection Environment), tripwire, integrit and samhain. Installing **debsums** will also help you to check the file system integrity, by comparing the md5sums of every file against the md5sums used in the Debian package archive. But beware: those files can easily be changed by an attacker and not all packages provide md5sums listings for the binaries they provided. For more information please read “Faça verificações de integridade periódicas” and “Fazendo um snapshot do sistema”.

You might want to use **locate** to index the whole filesystem, if so, consider the implications of that. The Debian findutils package contains **locate** which runs as user nobody, and so it only indexes files which are visible to everybody. However, if you change it's behaviour you will make all file locations visible to all users. If you want to index all the filesystem (not the bits that the user nobody can see) you can replace **locate** with the package slocate. slocate is labeled as a security enhanced version of GNU locate, but it actually provides additional file-locating functionality. When using **slocate**, the user only sees the actually accessible files and you can exclude any files or directories on the system. The slocate package runs its update process with higher privledges than locate, and indexes every file. Users are then able to quickly search for every file which they are able to see. **slocate** doesn't let them see new files; it filters the output based on your UID.

You might want to use bsign or elfsign. elfsign provides an utility to add a digital signature to an ELF binary and a second utility to verify that signature. The current implementation uses PKI to sign the checksum of the binary. The benefits of doing this are that it enables one to determine if a binary has been modified and who created it. bsign uses GPG, elfsign uses PKI (X.509) certificates (OpenSSL).

Configurando verificação de setuid

The Debian checksecurity package provides a **cron** job that runs daily in `/etc/cron.daily/checksecurity`[22]. This **cron** job will run the **/usr/sbin/checksecurity** script that will store information of this changes.

The default behavior does not send this information to the superuser but, instead keeps daily copies of the changes in `/var/log/setuid.changes`. You should set the MAILTO variable (in `/etc/checksecurity.conf`) to 'root' to have this information mailed to the superuser. See checksecurity(8) manual page for more configuration info.

[22] In previous releases, checksecurity was integrated into cron and the file was `/etc/cron.daily/standard`

Tornando o acesso a rede mais seguro

FIXME: More (Debian-specific) content needed.

Configurando características de rede do kernel

Many features of the kernel can be modified while running by echoing something into the `/proc` file system or by using **sysctl**. By entering **/sbin/sysctl** `-A` you can see what you can configure and what the options are, and it can be modified running

`/sbin/sysctl -w variable=value`

(see sysctl(8)). Only in rare cases do you need to edit something here, but you can increase security that way as well. For example:

```
net/ipv4/icmp_echo_ignore_broadcasts = 1
```

This is a *Windows emulator* because it acts like Windows on broadcast ping if this option is set to 1. That is, ICMP echo requests sent to the broadcast address will be ignored. Otherwise, it does nothing.

Se quer evitar que o seu sistema responda requisições ICMP, apenas ative esta opção de configuração:

```
net/ipv4/icmp_echo_ignore_all = 1
```

Para registrar pacotes com endereços impossíveis (devido a roteamento incorreto) em seu sistema, use:

```
/proc/sys/net/ipv4/conf/all/log_martians = 1
```

For more information on what things can be done with `/proc/sys/net/ipv4/*` read `/usr/src/linux/Documentation/filesystems/proc.txt`. All the options are described thoroughly under `/usr/src/linux/Documentation/networking/ip-sysctl.txt`[23].

Configuring syncookies

Esta opção é uma faca de dois gumes. De um lado ela protege o seu sistema contra flood de pacotes syn; por outro lado ela viola os padrões definidos (RFCs).

```
net/ipv4/tcp_syncookies = 1
```

If you want to change this option each time the kernel is working you need to change it in `/etc/network/options` by setting `syncookies=yes`. This will take effect when ever `/etc/init.d/networking` is run (which is typically done at boot time) while the following will have a one-time effect until the reboot:

```
echo 1 > /proc/sys/net/ipv4/tcp_syncookies
```

This option will only be available if the kernel is compiled with the `CONFIG_SYNCOOKIES`. All Debian kernels are compiled with this option builtin but you can verify it running:

[23] In Debian the kernel-source-`version` packages copy the sources to `/usr/src/kernel-source-version.tar.bz2`, just substitute `version` to whatever kernel version sources you have installed

```
$ sysctl -A |grep syncookies
net/ipv4/tcp_syncookies = 1
```

For more information on TCP syncookies read http://cr.yp.to/syncookies.html.

Tornando a rede segura em tempo de inicialização

Quando definir opções de configuração do kernel para a rede, você precisará configurá-la de forma que seja carregada sempre que o sistema for iniciado. O seguinte exemplo ativa muitas das opções anteriores assim como outras opções úteis.

There are actually two ways to configure your network at boot time. You can configure `/etc/sysctl.conf` (see: sysctl.conf(5)) or introduce a script that is called when the interface is enabled. The first option will be applied to all interfaces, whileas the second option allows you to configure this on a per-interface basis.

An example of a `/etc/sysctl.conf` configuration that will secure some network options at the kernel level is shown below. Notice the comment in it, `/etc/network/options` might override some values if they contradict those in this file when the `/etc/init.d/networking` is run (which is later than `procps` on the startup sequence).

```
#
# /etc/sysctl.conf - Configuration file for setting system variables
# See sysctl.conf (5) for information. Also see the files under
# Documentation/sysctl/, Documentation/filesystems/proc.txt, and
# Documentation/networking/ip-sysctl.txt in the kernel sources
# (/usr/src/kernel-$version if you have a kernel-package installed)
# for more information of the values that can be defined here.

#
# Be warned that /etc/init.d/procps is executed to set the following
# variables.  However, after that, /etc/init.d/networking sets some
# network options with builtin values.  These values may be overridden
# using /etc/network/options.
#
#kernel.domainname = example.com

# Additional settings - adapted from the script contributed
# by Dariusz Puchala (see below)
# Ignore ICMP broadcasts
net/ipv4/icmp_echo_ignore_broadcasts = 1
#
# Ignore bogus ICMP errors
net/ipv4/icmp_ignore_bogus_error_responses = 1
#
# Do not accept ICMP redirects (prevent MITM attacks)
net/ipv4/conf/all/accept_redirects = 0
# _or_
# Accept ICMP redirects only for gateways listed in our default
# gateway list (enabled by default)
# net/ipv4/conf/all/secure_redirects = 1
#
```

```
# Do not send ICMP redirects (we are not a router)
net/ipv4/conf/all/send_redirects = 0
#
# Do not forward IP packets (we are not a router)
# Note: Make sure that /etc/network/options has 'ip_forward=no'
net/ipv4/conf/all/forwarding = 0
#
# Enable TCP Syn Cookies
# Note: Make sure that /etc/network/options has 'syncookies=yes'
net/ipv4/tcp_syncookies = 1
#
# Log Martian Packets
net/ipv4/conf/all/log_martians = 1
#
# Turn on Source Address Verification in all interfaces to
# prevent some spoofing attacks
# Note: Make sure that /etc/network/options has 'spoofprotect=yes'
net/ipv4/conf/all/rp_filter = 1
#
# Do not accept IP source route packets (we are not a router)
net/ipv4/conf/all/accept_source_route = 0
```

To use the script you need to first create the script, for example, in `/etc/network/interface-secure` (the name is given as an example) and call it from `/etc/network/interfaces` like this:

```
auto eth0
iface eth0 inet static
        address xxx.xxx.xxx.xxx
        netmask 255.255.255.xxx
        broadcast xxx.xxx.xxx.xxx
        gateway xxx.xxx.xxx.xxx
        pre-up /etc/network/interface-secure
```

In this example, before the interface eth0 is enabled the script will be called to secure all network interfaces as shown below.

```
#!/bin/sh -e
# Script-name: /etc/network/interface-secure
#
# Modifies some default behavior in order to secure against
# some TCP/IP spoofing & attacks for all interfaces.
#
# Contributed by Dariusz Puchalak.
#
echo 1 > /proc/sys/net/ipv4/icmp_echo_ignore_broadcasts
                                        # Broadcast echo protection enabled.
echo 0 > /proc/sys/net/ipv4/conf/all/forwarding
                                        # IP forwarding disabled.
echo 1 > /proc/sys/net/ipv4/tcp_syncookies # TCP syn cookies protection enabled.
echo 1 >/proc/sys/net/ipv4/conf/all/log_martians # Log strange packets.
# (this includes spoofed packets, source routed packets, redirect packets)
# but be careful with this on heavy loaded web servers.
echo 1 > /proc/sys/net/ipv4/icmp_ignore_bogus_error_responses
```

```
                                                # Bad error message protection enabled.

# IP spoofing protection.
echo 1 > /proc/sys/net/ipv4/conf/all/rp_filter

# Disable ICMP redirect acceptance.
echo 0 > /proc/sys/net/ipv4/conf/all/accept_redirects
echo 0 > /proc/sys/net/ipv4/conf/all/send_redirects

# Disable source routed packets.
echo 0 > /proc/sys/net/ipv4/conf/all/accept_source_route

exit 0
```

Notice that you can actually have per-interface scripts that will enable different network options for different interfaces (if you have more than one), just change the pre-up line to:

```
pre-up /etc/network/interface-secure $IFACE
```

And use a script which will only apply changes to a specific interface, not to all of the interfaces available. Notice that some networking options can only be enabled globally, however. A sample script is this one:

```
#!/bin/sh -e
# Script-name: /etc/network/interface-secure
#
# Modifies some default behavior in order to secure against
# some TCP/IP spoofing & attacks for a given interface.
#
# Contributed by Dariusz Puchalak.
#

IFACE=$1
if [ -z "$IFACE" ] ; then
   echo "$0: Must give an interface name as argument!"
   echo "Usage: $0 <interface>"
   exit 1
fi

if [ ! -e /proc/sys/net/ipv4/conf/$IFACE/ ]; then
   echo "$0: Interface $IFACE does not exit (cannot find /proc/sys/net/ipv4/conf/)
   exit 1
fi

echo 0 > /proc/sys/net/ipv4/conf/$IFACE/forwarding  # IP forwarding disabled.
echo 1 >/proc/sys/net/ipv4/conf/$IFACE/log_martians # Log strange packets.
# (this includes spoofed packets, source routed packets, redirect packets)
# but be careful with this on heavy loaded web servers.

# IP spoofing protection.
echo 1 > /proc/sys/net/ipv4/conf/$IFACE/rp_filter

# Disable ICMP redirect acceptance.
echo 0 > /proc/sys/net/ipv4/conf/$IFACE/accept_redirects
```

```
echo 0 > /proc/sys/net/ipv4/conf/$IFACE/send_redirects

# Disable source routed packets.
echo 0 > /proc/sys/net/ipv4/conf/$IFACE/accept_source_route

exit 0
```

An alternative solution is to create an `init.d` script and have it run on bootup (using **update-rc.d** to create the appropriate `rc.d` links).

Configurando características do firewall

In order to have firewall capabilities, either to protect the local system or others *behind* it, the kernel needs to be compiled with firewall capabilities. The standard Debian 2.2 kernel (Linux 2.2) provides the packet filter **ipchains** firewall, Debian 3.0 standard kernel (Linux 2.4) provides the *stateful* packet filter **iptables** (netfilter) firewall.

In any case, it is pretty easy to use a kernel different from the one provided by Debian. You can find pre-compiled kernels as packages you can easily install in the Debian system. You can also download the kernel sources using the kernel-source-`X` and build custom kernel packages using **make-kpkg** from the kernel-package package.

Setting up firewalls in Debian is discussed more thoroughly in “Adicionando capacidades de firewall”.

Desativando assuntos relacionados a weak-end de máquinas

Systems with more than one interface on different networks can have services configured so that they will bind only to a given IP address. This usually prevents access to services when requested through any other address. However, this does not mean (although it is a common misconception) that the service is bound to a given *hardware* address (interface card). [24]

It seems, however, not to work with services bound to 127.0.0.1, you might need to write the tests using raw sockets.

This is not an ARP issue and it's not an RFC violation (it's called *weak end host* in RFC1122 [ftp://ftp.isi.edu/in-notes/rfc1122.txt], (in the section 3.3.4.2). Remember, IP addresses have nothing to do with physical interfaces.

Nos kernels da série 2.2 (e anteriores) isto pode ser corrigido com:

```
# echo 1 > /proc/sys/net/ipv4/conf/all/hidden
```

[24] To reproduce this (example provided by Felix von Leitner on the Bugtraq mailing list):

```
host a (eth0 connected to eth0 of host b):
  ifconfig eth0 10.0.0.1
  ifconfig eth1 23.0.0.1
  tcpserver -RHl localhost 23.0.0.1 8000 echo fnord

host b:
  ifconfig eth0 10.0.0.2
  route add 23.0.0.1 gw 10.0.0.1
  telnet 23.0.0.1 8000
```

```
# echo 1 > /proc/sys/net/ipv4/conf/eth0/hidden
# echo 1 > /proc/sys/net/ipv4/conf/eth1/hidden
.....
```

Em outros kernels, isto pode ser corrigido com uma das alternativas:

- regras do iptables.
- properly configured routing. [25]
- kernel patching. [26]

Along this text there will be many occasions in which it is shown how to configure some services (sshd server, apache, printer service...) in order to have them listening on any given address, the reader should take into account that, without the fixes given here, the fix would not prevent accesses from within the same (local) network. [27]

FIXME: Comments on Bugtraq indicate there is a Linux specific method to bind to a given interface.

FIXME: Enviar um bug contra o netbase, assim a correção de roteamento será o comportamento padrão no Debian?

Protegendo-se contra ataques ARP

Quando não confia em outras máquinas na sua rede (que deve sempre ser o caso, por ser uma atitude mais segura) você deverá proteger a si mesmo de vários ataques ARP existentes.

As you know the ARP protocol is used to link IP addresses to MAC addresses (see ftp://ftp.isi.edu/in-notes/rfc826.txt for all the details). Every time you send a packet to an IP address an ARP resolution is done (first by looking into the local ARP cache then if the IP isn't present in the cache by broadcasting an ARP query) to find the target's hardware address. All the ARP attacks aim to fool your box into thinking that box B's IP address is associated to the intruder's box's MAC address; Then every packet that you want to send to the IP associated to box B will be send to the intruder's box...

Those attacks (ARP cache poisoning, ARP spoofing...) allow the attacker to sniff the traffic even on switched networks, to easily hijack connections, to disconnect any host from the network... ARP attacks are powerful and simple to implement, and several tools exists, such as **arpspoof** from the dsniff package or http://arpoison.sourceforge.net/.

No entanto, sempre existe uma solução:

- Use a static ARP cache. You can set up "static" entries in your ARP cache with:

[25] The fact that this behavior can be changed through routing was described by Matthew G. Marsh in the Bugtraq thread:

```
eth0 = 1.1.1.1/24
eth1 = 2.2.2.2/24

ip rule add from 1.1.1.1/32 dev lo table 1 prio 15000
ip rule add from 2.2.2.2/32 dev lo table 2 prio 16000

ip route add default dev eth0 table 1
ip route add default dev eth1 table 2
```

[26] There are some patches available for this behavior as described in Bugtraq's thread at http://www.linuxvirtualserver.org/~julian/#hidden and http://www.fefe.de/linux-eth-forwarding.diff.

[27] An attacker might have many problems pulling the access through after configuring the IP-address binding while not being on the same broadcast domain (same network) as the attacked host. If the attack goes through a router it might be quite difficult for the answers to return somewhere.

```
arp -s host_name hdwr_addr
```

By setting static entries for each important host in your network you ensure that nobody will create/modify a (fake) entry for these hosts (static entries don't expire and can't be modified) and spoofed ARP replies will be ignored.

- Detect suspicious ARP traffic. You can use arpwatch, karpski or more general IDS that can also detect suspicious ARP traffic (snort, http://www.prelude-ids.org...).
- Implementando filtragem na validação de tráfego IP no endereço MAC.

Fazendo um snapshot do sistema

Before putting the system into production system you could take a snapshot of the whole system. This snapshot could be used in the event of a compromise (see Capítulo 11, *Depois do comprometimento do sistema (resposta a incidentes)*). You should remake this upgrade whenever the system is upgraded, especially if you upgrade to a new Debian release.

For this you can use a writable removable-media that can be set up read-only, this could be a floppy disk (read protected after use), a CD on a CD-ROM unit (you could use a rewritable CD-ROM so you could even keep backups of md5sums in different dates), or a USB disk or MMC card (if your system can access those and they can be write protected).

O seguinte script criará o snapshot:

```
#!/bin/bash
/bin/mount /dev/fd0 /mnt/floppy
trap "/bin/umount /dev/fd0" 0 1 2 3 9 13 15
if [ ! -f /usr/bin/md5sum ] ; then
 echo "Cannot find md5sum. Aborting."
 exit 1
fi
/bin/cp /usr/bin/md5sum /mnt/floppy
echo "Calculating md5 database"
>/mnt/floppy/md5checksums.txt
for dir in /bin/ /sbin/ /usr/bin/ /usr/sbin/ /lib/ /usr/lib/
do
   find $dir -type f | xargs /usr/bin/md5sum >>/mnt/floppy/md5checksums-lib.txt
done
echo "post installation md5 database calculated"
if [ ! -f /usr/bin/sha1sum ] ; then
 echo "Cannot find sha1sum"
        echo "WARNING: Only md5 database will be stored"
else
 /bin/cp /usr/bin/sha1sum /mnt/floppy
 echo "Calculating SHA-1 database"
 >/mnt/floppy/sha1checksums.txt
 for dir in /bin/ /sbin/ /usr/bin/ /usr/sbin/ /lib/ /usr/lib/
 do
    find $dir -type f | xargs /usr/bin/sha1sum >>/mnt/floppy/sha1checksums-lib.txt
 done
```

```
 echo "post installation sha1 database calculated"
fi
exit 0
```

Note that the md5sum binary (and sha1sum, if available) is placed on the floppy drive so it can be used later on to check the binaries of the system (just in case it gets trojaned). However, if you want to make sure that you are running a legitimate binary, you might want to either compile a static copy of the md5sum binary and use that one (to prevent a trojaned libc library from interfering with the binary) or to use the snapshot of md5sums only from a clean environment such as a rescue CD-ROM or a Live-CD (to prevent a trojaned kernel from interfering). I cannot stress this enough: if you are on a compromised system you cannot trust its output, see Capítulo 11, *Depois do comprometimento do sistema (resposta a incidentes)*.

The snapshot does not include the files under `/var/lib/dpkg/info` which includes the MD5 hashes of installed packages (in files ending with `.md5sums`). You could copy this information along too, however you should notice:

- the md5sums files include the md5sum of all files provided by the Debian packages, not just system binaries. As a consequence, that database is bigger (5 Mb versus 600 Kb in a Debian GNU/Linux system with a graphical system and around 2.5 Gb of software installed) and will not fit in small removable media (like a single floppy disk, but would probably fit in a removable USB memory).

- not all Debian packages provide md5sums for the files installed since it is not (currently) mandated policy. Notice, however, that you can generate the md5sums for all packages using debsums after you've finished the system installation:

```
# debsums --generate=missing,keep
```

Once the snapshot is done you should make sure to set the medium read-only. You can then store it for backup or place it in the drive and use it to drive a **cron** check nightly comparing the original md5sums against those on the snapshot.

If you do not want to setup a manual check you can always use any of the integrity systems available that will do this and more, for more information please read “Faça verificações de integridade periódicas”.

Outras recomendações

Não use programas que dependem da svgalib

SVGAlib is very nice for console lovers like me, but in the past it has been proven several times that it is very insecure. Exploits against **zgv** were released, and it was simple to become root. Try to prevent using SVGAlib programs wherever possible.

Capítulo 5. Tornando os serviços em execução do seu sistema mais seguros

Os serviços podem ser deixados mais seguros de duas formas:

- Tornando-os somente acessíveis em pontos de acessos (interfaces) que são utilizados.
- Configurando-os adequadamente, desta forma eles poderão somente ser usados por usuários legítimos de forma autorizada.

A restrição de serviços de forma que possam somente ser acessados de um determinado lugar pode ser feito restringindo o acesso a eles no nível de kernel (i.e. firewall), configure-os para operar somente em interfaces definidas (alguns serviços podem não ter esta característica) ou usando algum outro método, por exemplo o patch vserver do Linux (para 2.4.16) pode ser usado para forçar o kernel a utilizar somente uma interface de rede.

Regarding the services running from **inetd** (**telnet**, **ftp**, **finger**, **pop3**...) it is worth noting that **inetd** can be configured so that services only listen on a given interface (using `service@ip` syntax) but that's an undocumented feature. One of its substitutes, the **xinetd** meta-daemon includes a `bind` option just for this matter. See ixnetd.conf(5) manual page.

```
service nntp
{
        socket_type     = stream
        protocol        = tcp
        wait            = no
        user            = news
        group           = news
        server          = /usr/bin/env
        server_args     = POSTING_OK=1 PATH=/usr/sbin/:/usr/bin:/sbin/:/bin
+/usr/sbin/snntpd logger -p news.info
        bind            = 127.0.0.1
}
```

As seguintes seções detalham como alguns serviços individuais podem ser configurados adequadamente conforme sua utilização.

Tornando o ssh mais seguro

Caso ainda estiver usando o telnet ao invés do ssh, você deverá dar uma parada na leitura deste manual e alterar isto. O ssh deve ser usado para qualquer login remoto ao invés do telnet. Em uma era onde é fácil capturar o tráfego que circula na internet e obter senhas em texto plano, você deverá usar somente protocolos que utilizam criptografia. Assim, execute um `apt-get install ssh` agora em seu sistema.

Encoraje todos os usuários em seu sistema para utilizarem o ssh ao invés do telnet, ou até mesmo melhor, remova o telnet/telnetd. Em adição, você deverá evitar entrar no sistema usando o ssh como usuário root e ao invés disto, usar métodos alternativos para se tornar o root, como o **su** ou **sudo**. Finalmente, o arquivo `sshd_config` no diretório `/etc/ssh`, também deverá ser modificado para aumentar a segurança:

- Especifica que o ssh somente funcionará na interface especificada, caso tenha mais de uma interface (e não deseja que o ssh funcione através delas) ou em caso de adição de uma futura interface de rede (onde não deseja receber conexões ssh através dela).

- Tenta não permitir o login do usuário Root sempre que possível. Se alguém quiser se tornar o usuário root usando ssh, agora dois logins são necessários e o ataque de força bruta não terá efeito no root via SSH.
- `Port 666` or `ListenAddress 192.168.0.1:666` Change the listen port, so the intruder cannot be completely sure whether a sshd daemon runs (be forewarned, this is security by obscurity).
- `PermitEmptyPasswords no` Empty passwords make a mockery of system security.
- Permite somente certos usuários terão acesso via ssh a esta maquina. `usuario@maquina` pode também ser usado para restringir um determinado usuário de acessar somente através de uma maquina especificada.
- Permite somente membros de certos grupos de terem acesso ao ssh nesta maquina. AllowGroups e AllowUsers possuem diretivas equivalentes para bloquear o acesso a maquina. Não se surpreenda por eles serem chamados de "DenyUsers" e "DenyGroups".
- Esta escolha fica completamente por sua conta. É mais seguro somente permitir o acesso a maquina de usuários com chaves ssh colocadas em `~/.ssh/authorized_keys`. Se deseja isto, ajuste esta opção para "no".
- Disable any form of authentication you do not really need, if you do not use, for example `RhostsRSAAuthentication`, `HostbasedAuthentication`, `KerberosAuthentication` or `RhostsAuthentication` you should disable them, even if they are already by default (see the manpage sshd_config(5) manual page).
- Desative o protocolo da versão 1, pois ele tem alguns problemas de design que torna fácil a descoberta de senhas. Para mais informações leia http://earthops.net/ssh-timing.pdf ou o http://xforce.iss.net/static/6449.php.
- Adiciona um banner (ele será lido de um arquivo) para usuários se conectando ao servidor ssh, em alguns países o envio de avisos antes de acessar um determinado sistema alertando sobre acesso não autorizado ou monitoramento de usuários deverá ser emitido para ter proteção legal.

Você também poderá restringir o acesso ao servidor ssh usando o `pam_listfile` ou `pam_wheel` no arquivo de controle PAM para o ssh restringir os logins ssh. Por exemplo, se quiser manter qualquer pessoa não listada em `/etc/loginusers` adicionando esta linha no `/etc/pam.d/ssh`:

```
auth       required     pam_listfile.so sense=allow onerr=fail item=user file=/etc
```

Como nota final, tenha atenção que estas diretivas são válidas para um arquivo de configuração do OpenSSH. Atualmente, não freqüentemente usados três tipos de implementações conhecidas do daemon: ssh1, ssh2 e OpenSSH feito pelo time do OpenBSD. O ssh1 foi o primeiro daemon disponível e é ainda o mais usado (existem rumores que até existe um porte para Windows). O ssh2 possui mais vantagens sobre o ssh2, exceto que ele é lançado sob uma licença fonte fechado. O OpenSSH é um daemon ssh completamente livre, que suporta ambos os protocolos ssh1 e ssh2. O OpenSSH é a versão instalada junto o Debian quando o pacote ssh é escolhido.

You can read more information on how to set up SSH with PAM support in the http://lists.debian.org/debian-security/2001/11/msg00395.html.

Executando o ssh em uma jaula chroot

O OpenSSH atualmente não suporta um método de chroot automático durante a conexão do usuário (a versão comercial oferece esta funcionalidade). No entanto existe um projeto para fornecer esta funcionalidade também para o ssh, veja http://chrootssh.sourceforge.net, atualmente ele não esta

empacotado para o Debian. Você poderá usar, no entanto, o módulo `pam_chroot` como descrito em “Restringindo acessos de usuários”.

Em “Chroot environment for SSH” você terá diversas opções para criar um ambiente chroot para o SSH.

Clientes do ssh

Se estiver usando um cliente SSH com um servidor SSH, você deverá ter certeza que ele suporta os mesmos protocolos que são especificados no servidor. Por exemplo, se utilizar o pacote mindterm, ele somente utiliza a versão 1 . No entanto, o servidor ssh utiliza, por padrão, a configuração para aceitar somente conexões para o protocolo da versão 2 (por razões de segurança).

Desativando transferências de arquivos

If you do *not* want users to transfer files to and from the ssh server you need to restrict access to the **sftp-server***and* the **scp** access. You can restrict **sftp-server** by configuring the proper `Subsystem` in the `/etc/ssh/sshd_config`.

You can also chroot users (using libpam-chroot so that, even if file transfer is allowed, they are limited to an environment which does not include any system files.

Restricing access to file transfer only

You might want to restrict access to users so that they can only do file transfers and cannot have interactive shells. In order to do this you can either:

- bloquear o login de usuários ao servidor ssh (como descrito acima no arquivo de configuração ou configuração do PAM).
- give users a restricted shell such as scponly or rssh. These shells restrict the commands available to the users so that they are not provided any remote execution priviledges.

Tornando o Squid mais seguro

Squid is one of the most popular proxy/cache server, and there are some security issues that should be taken into account. Squid's default configuration file denies all users requests. However the Debian package allows access from 'localhost', you just need to configure your browser properly. You should configure Squid to allow access to trusted users, hosts or networks defining an Access Control List on `/etc/squid/squid.conf`, see the https://web.archive.org/web/20061206052115/http://www.deckle.co.za/squid-users-guide/Main_Page for more information about defining ACLs rules. Notice that Debian provides a minimum configuration for Squid that will prevent anything, except from *localhost* to connect to your proxy server (which will run in the default port 3128). You will need to customize your `/etc/squid/squid.conf` as needed.

The recommended minimum configuration (provided with the package) is shown below:

```
acl all src 0.0.0.0/0.0.0.0
acl manager proto cache_object
acl localhost src 127.0.0.1/255.255.255.255
acl SSL_ports port 443 563
acl Safe_ports port 80          # http
acl Safe_ports port 21          # ftp
```

```
acl Safe_ports port 443 563     # https, snews
acl Safe_ports port 70          # gopher
acl Safe_ports port 210         # wais
acl Safe_ports port 1025-65535  # portas não registradas
acl Safe_ports port 280         # http-mgmt
acl Safe_ports port 488         # gss-http
acl Safe_ports port 591         # filemaker
acl Safe_ports port 777         # multiling http
acl Safe_ports port 901         # SWAT
acl purge method PURGE
acl CONNECT method CONNECT
(...)
# Somente permite acesso do cachemgr vindos de localhost
http_access allow manager localhost
http_access deny manager
# Somente permite requisições de purge vindas de localhost
http_access allow purge localhost
http_access deny purge
# Bloqueia requisições para portas desconhecidas
http_access deny !Safe_ports
# Bloqueia CONNECT a portas que não sejam SSL
http_access deny CONNECT !SSL_ports
#
# INSIRA SUAS PRÓPRIAS REGRAS AQUI PARA PERMITIR O ACESSO DE SEUS CLIENTE
#
http_access allow localhost
# E finalmente bloqueia qualquer outro acesso a este proxy
http_access deny all
#Padrão:
# icp_access deny all
#
#Permite requisições ICQ vindas de qualquer pessoa
icp_access allow all
```

Você também deverá configurar o Squid baseado nos recursos do seu sistema, incluindo a memória cache (opção `cache_mem`), localização dos arquivos de cache e quantidade de espaço que utilizarão no disco (opção `cache_dir`).

Note que, se não for corretamente configurado, alguém poderá enviar mensagens de e-mail através do squid, pois os protocolos HTTP e SMTP tem design similar. O arquivo de configuração padrão do Squid bloqueia o acesso a porta 25. Se desejar permitir conexões a porta 25, apenas adicione-a a lista Safe_ports. No entanto, isto *NÃO* é recomendado.

Ajustar e configurar um servidor proxy/cache é apenas parte da tarefa de manter um site seguro. Outra tarefa necessária é a análise dos logs do Squid para ter certeza que todas as coisas estão funcionando como deveriam estar. Existem alguns pacotes no Debian GNU/Linux que podem ajudar o administrador a fazer isto. Os seguintes pacotes estão disponíveis na woody (Debian 3.0):

- calamaris - Analisador de arquivos de log para o Squid ou log do proxy Oops
- modlogan - Um analisador de arquivos e log modular.
- sarg - Squid Analysis Report Generator.
- squidtaild - Programa de monitoramento de logs do Squid.

When using Squid in Accelerator Mode it acts as a web server too. Turning on this option increases code complexity, making it less reliable. By default Squid is not configured to act as a web server, so you don't need to worry about this. Note that if you want to use this feature be sure that it is really necessary. To find more information about Accelerator Mode on Squid see the https://web.archive.org/web/20070104164802/http://www.deckle.co.za/squid-users-guide/Accelerator_Mode

Tornando o FTP mais seguro

Se realmente precisar usar o FTP (sem transportá-lo com sslwrap ou dentro de um tunel SSL ou SSH), você deverá fazer um chroot dentro do diretório de usuários do ftp, assim o usuário será incapaz de ver qualquer coisa que não seja seu próprio diretório. Caso contrário, ele poderá atravessar seu sistema de arquivos raíz como se tivesse uma conta shell. Você poderá adicionar a seguinte linha no seu arquivo `proftpd.conf` na sua seção global para ativar esta característica chroot:

```
DefaultRoot ~
```

Reinicie o proftpd executando `/etc/init.d/proftpd restart` e verifique se agora pode escapar do seu diretório de usuário.

Para prevenir ataques DoS usando ../../.., adicione a seguinte linha no seu arquivo `/etc/proftpd.conf`: `DenyFilter \*.*/`

Lembre-se sempre que o FTP envia o login e senhas de autenticação em texto plano (isto não é um problema se estiver oferecendo acesso a serviços públicos. Entretanto existem alternativas melhores no Debian para isto, como o **sftp** (fornecido pelo pacote ssh). Também existem implementações livres do ssh para outros sistemas operacionais, por exemplo: http://www.chiark.greenend.org.uk/~sgtatham/putty/ e o http://www.cygwin.com.

No entanto, se você ainda mantém um servidor FTP enquanto disponibiliza o acesso através do SSH você deve encontrar um problema típico. Usuários acessando servidores FTP anônimos dentro de sistemas protegidos com o SSH devem tentar efetuar o login no *FTP server*. Enquanto o acesso será recusado, as senhas nunca serão enviadas na rede de forma desprotegida. Para evitar isto, o desenvolvedor TJ Sauders do ProFTPd , criou um patch que evita que os usuários utilizem um servidor FTP anônimo com uma conta válida do ssh. Mais informações e o patch estão disponíveis em: http://www.castaglia.org/proftpd/#Patches. Este patch também foi reportado para o Debian, veja http://bugs.debian.org/145669.

Tornando o acesso ao sistema X Window mais seguro

Hoje em dia, terminais do X são usados por mais e mais empresas onde é necessário para várias estações de trabalho. Isto pode ser perigoso, porque você precisa permitir o servidor de arquivos a se conectar aos clientes (a partir do ponto de vista do servidor X, o X altera a definição de cliente e servidor). Se você seguir a (péssima) sugestão de muitas documentações você digitará `xhost +` em sua máquina. Isto permitirá qualquer cliente do X a se conectar em seu sistema. Para ter um pouco mais de segurança, você deverá usar o comando `xhost +hostname` ao invés de somente permitir acessos através de máquinas específicas.

Uma solução muito mais segura, no entanto, é usar o ssh para para toda a seção. Isto é feito automaticamente quando você faz um ssh para a outra máquina. Para isto funcionar, você terá que configurar ambos o cliente ssh e o servidor ssh. No cliente ssh, a opção `ForwardX11` deverá estar ajustada para `yes` no arquivo `/etc/ssh/ssh_config`. No servidor ssh, a opção `X11Forwarding` deverá estar ajustada para `yes` no arquivo `/etc/ssh/sshd_config` e o pacote xbase-clients deverá estar

instalado, pois o servidor ssh utiliza o `/usr/X11R6/bin/xauth` quando está configurando uma tela de pseudo terminal do X. Nos tempos do SSH, agora você deverá deixar de usar o controle de acesso baseado em xhost completamente.

Para melhor segurança, você não precisará permitir o acesso ao X a partir de outras máquinas, isto é feito desativando o servidor na porta 6000 simplesmente digitando:

```
$ startx -- -nolisten tcp
```

Este é o comportamento padrão do Xfree 4.1.0 (o Xserver fornecido no Debian 3.0). Se estiver executando o Xfree 3.3.6 (i.e. você tem o Debian 2.2 instalada) você poderá editar o arquivo `/etc/X11/xinit/xserverrcc` e fazer a alteração nestas seguintes linhas:

```
#!/bin/sh
exec /usr/bin/X11/X -dpi 100 -nolisten tcp
```

If you are using XDM set `/etc/X11/xdm/Xservers` to: `:0 local /usr/bin/X11/X vt7 -dpi 100 -nolisten tcp`. If you are using Gdm make sure that the `DisallowTCP=true` option is set in the `/etc/gdm/gdm.conf` (which is the default in Debian). This will basically append `-nolisten tcp` to every X command line [1].

Você também poderá configurar o timeout padrão para o travamento do **xscreensaver**. Até mesmo se o usuário substituir este valor, você poderá editar o arquivo `/etc/X11/app-defaults/XScreenSaver` e alterar a linha:

```
*lock:                  False
```

(que é padrão no Debian) para:

```
*lock:                  True
```

FIXME: adicionar informações sobre como desativar as proteções de tela que mostra o desktop do usuário (que pode conter informações sensíveis).

Leia mais sobre a segurança em servidores X Window em http://www.tldp.org/HOWTO/XWindow-User-HOWTO.html (`/usr/share/doc/HOWTO/en-txt/XWindow-User-HOWTO.txt.gz`).

FIXME: Adicionar informações sobre a discussão na debian-security sobre como alterar os arquivos de configuração no servidor XFree 3.3.6 para fazer isto.

Verifique seu gerenciador de tela

Se somente quiser ter um gerenciador de tela instalado para uso local (tendo um lindo login gráfico) tenha certeza que tudo que estiver relacionado com o XDMCP (X Display Manager Control Protocol) está desativado. No XDM você poderá fazer isto através da linha em `/etc/X11/xdm/xdm-config`:

```
DisplayManager.requestPort:     0
```

For GDM there should be in your gdm.conf:

[1] Gdm will *not* append `-nolisten tcp` if it finds a `-query` or `-indirect` on the command line since the query wouldn't work.

```
[xdmcp]
Enable=false
```

Normalmente, todos os gerenciadores de tela estão configurados para não iniciar serviços do XDMCP por padrão no Debian.

Tornando o servidor de impressão mais seguro (sobre o lpd e lprng)

Imagine, você chegando ao trabalho e a impressora jogando fora uma quantidade impressionante de papel porque alguém esta fazendo um DoS em seu daemon de impressão. Desagradável, não é?

Em qualquer arquitetura de impressão do unix, deverá existir uma forma de enviar os dados do cliente para o servidor de impressão. No tradicional **lpr** e **lp**, os comandos do cliente copiam ou fazem um link simbólico de dados no diretório de spool (este é o motivo porque estes programas normalmente são SUID ou SGID).

Para evitar quaisquer anormalidade, você deverá manter o seu servidor de impressão especialmente seguro. Isto significa que precisa configurar seu serviço de impressão de forma que só permita conexões de um conjunto de máquinas confiáveis. Para fazer isto, adicione os servidores que deseja permitir a impressão em seu arquivo `/etc/hosts.lpd`.

No entanto, até mesmo se fizer isto, o **lpr** aceitará conexões de entrada na porta 515 de qualquer interface. Você deverá considerar fazer um firewall das conexões de redes/hosts que não tenham permissão de impressão (o daemon **lpr** não tem a possibilidade de aceitar conexões em somente um determinado endereço IP).

O **Lprng** deverá ser o preferido em cima do **lpr** pois ele pode ser configurado para fazer controle de acesso por IP. E você poderá especificar qual interface escutará por conexões (embora algumas vezes pareça um pouco estranho).

Se utilizar uma impressora em seu sistema, mas somente localmente, você não desejará compartilhar este serviço através de uma rede. Você poderá considerar o uso de outros sistemas de impressão, tal como o fornecido pelo pacote cups ou pelo http://pdq.sourceforge.net/ que é baseado em permissões do usuário no dispositivo `/dev/lp0`.

No cups, os dados de impressão são transferidos ao servidores via protocolo http. Isto significa que o programa cliente não precisa de qualquer privilégio especial, mas requer que o servidor escute em uma porta, em algum lugar.

No entanto, se quiser usar o **cups**, mas somente localmente, você poderá configura-lo para escutar na interface loopback alterando o arquivo de configuração `/etc/cups/cupsd.conf`:

```
Listen 127.0.0.1:631
```

Existem muitas outras opções de segurança como permitir ou bloquear redes e máquinas neste arquivo de configuração. No entanto, se você não precisar delas, será melhor que limite simplesmente a porta onde o programa espera por conexões. O **Cups** também serve documentações através da porta HTTP. Se não quiser revelar informações úteis em potencial para invasores externos também adicione:

```
<Location />
  Order Deny,Allow
   Deny From All
    Allow From 127.0.0.1
</Locationi>
```

Este arquivo de configuração pode ser modificado para adicionar algumas outras características incluindo certificados SSL/TLS e criptografia. Os manuais estão disponíveis em http://localhost:631/ ou em cups.org.

FIXME: Adicionar mais conteúdo (o artigo em http://www.rootprompt.org fornecendo visões mais interessantes).

FIXME: Verificar se o PDG está disponível no Debian, e se estiver, sugerir como sistema de impressão preferido.

FIXME: Verificar se o Farmer/Wietse possui um substituto para daemon de impressão e se está disponível no Debian.

Tornando o serviço de e-mails seguro

Se seu servidor não for um servidor de mensagens, e realmente não precisa ter um programa esperando por conexões de entradas, mas deseja que as mensagens locais sejam entregues, por exemplo, para recebimento de mensagens do usuário root de qualquer alerta de segurança que tenha no local.

Se tiver o **exim** você não precisará do daemon funcionando para fazer isto, pois o pacote padrão do **cron** esvazia a fila de mensagens. Veja “Desabilitando daemons de serviço” para saber como fazer isto.

Configurando um programa de e-mails nulo

Você pode querer ter um daemon de mensagens locais assim ele poderá repassar os e-mails enviados localmente para outro sistema. Isto é comum quando você tem que administrar um número de máquinas e não quer conectar a cada uma delas para ler as mensagens enviadas localmente. Assim como todos os logs de cada sistema individual podem ser centralizados usando um servidor de logs central, as mensagens podem ser enviadas para um servidor de mensagens central.

Tal sistema *somente-repasse* deverá ser configurado adequadamente para fazer isto. O daemon poderá, também, ser configurado para somente esperar por conexões no endereço de loopback.

The following configuration steps only need to be taken to configure the exim package in the Debian 3.0 release. If you are using a later release (such as 3.1 which uses exim4) the installation system has been improved so that if the mail transport agent is configured to only deliver local mail it will automatically only allow connections from the local host and will not permit remote connections.

In a Debian 3.0 system using exim, you will have to remove the SMTP daemon from **inetd**:

```
$ update-inetd --disable smtp
```

e configurar o daemon de mensagens para somente esperar por conexões na interface loopback. No **exim** (o MTA padrão) você poderá fazer isto editando o arquivo de configuração `/etc/exim.conf` e adicionando a seguinte linha:

```
local_interfaces = "127.0.0.1"
```

Reinicie ambos os daemons (inetd e exim) e você terá o exim esperando por conexões somente no soquete 127.0.0.1:25. Seja cauteloso e desative primeiro o inetd, caso contrário, o exim não iniciará pois o daemon do inetd já está esperando por conexões de entrada.

Para o **postfix**, edite o arquivo `/etc/postfix/main.conf`:

```
inet_interfaces = localhost
```

Se quiser somente mensagens locais, este método é melhor que utilizar o método tcp wrappers no daemon de mensagens ou adicionar regras de firewall para que ninguém acesse-o. No entanto, se precisar que ele escute em outras interfaces, você deverá considerar carrega-lo a partir do inetd e adicionar um tcp wrapper, assim as conexões de entradas são verificadas nos arquivos `/etc/hosts.allow` e `/etc/hosts.deny`. Também, você deverá estar atento sobre acessos não autorizados sendo tentados sobre o seu daemon de mensagens, se configurar adequadamente o log de mensagens do seu sistema para qualquer um dos métodos acima.

Em qualquer caso, para rejeitar tentativas de repasse de mensagens a nível SMTP, você deverá alterar o arquivo `/etc/exim/exim.conf` para incluir:

```
receiver_verify = true
```

Até mesmo se seu servidor de e-mails não repassar a mensagem, este tipo de configuração é necessário para o teste de relay em http://www.abuse.net/relay.html para determinar que seu servidor *não* é capaz de repassar mensagens.

No entanto, se desejar uma configuração somente de leitura, você poderá considerar a alteração do daemon de mensagens para programas que podem *somente* ser configurados para redirecionar as mensagens para servidores de mensagens remotas. O Debian atualmente oferece o pacote ssmtp e o nullmailer para este propósito. Em qualquer caso, você deverá avaliar por si mesmo quaisquer dos agentes de transporte de mensagens [2] fornecido com o Debian. Veja que programa atende melhor aos propósitos do sistema.

Fornecendo acesso seguro às caixas de mensagens

Se quiser oferecer acesso remoto às caixas de mensagens, existe um número de daemons POP3 e IMAP disponíveis [3] . No entanto, se você oferecer acesso a IMAP, note que ele é um protocolo de acesso a arquivos, ele pode se tornar equivalente a um acesso shell porque os usuários podem ser capazes de obter qualquer arquivo através dele.

Tente, por exemplo, configurar como seu caminho para a inbox`{servidor.com}/etc/passwd`, se ele abrir o arquivo com sucesso seu daemon IMAP não está corretamente configurado para prevenir este tipo de acesso.

Dos servidores de IMAP existentes no Debian, o servidor **cyrus** (do pacote cyrus-imapd) contorna isto tendo todos os acessos sendo em um banco de dados mantido em uma parte restrita do sistema de arquivos.

[2] para obter uma lista de todos os daemons de mensagens disponíveis no Debian, execute o comando:

```
$ apt-cache search mail-transport-agent
```

A lista não incluirá o **qmail**, que é distribuído somente como código fonte no pacote qmail-src.

[3] Uma lista de servidores/daemons que suportam estes protocolos podem ser obtidos com:

```
$ apt-cache search pop3-server $ apt-cache search imap-server
```

Também o **uw-imapd** (ou instale o uw-imapd ou melhor, se seus clientes IMAP o suportam, uw-imapd-ssl) poderá ser configurado para fazer o chroot do diretório dos usuários de mensagens mas isto não é ativado por padrão. A documentação fornecida oferece mais informações sober como configura-lo.

Também, você pode tentar executar um servidor IMAP que não precisa de usuários válidos sendo criados no sistema local (que também oferece acesso a shell). Ambos os pacotes courier-imap (para IMAP) e courier-pop teapop (para o POP3) e o cyrus-imapd (para ambos POP3 e IMAP) fornecem servidores com métodos de autenticação que não dependem de contas locais de usuários. O **cyrus** pode usar qualquer método de autenticação que possa ser configurado através do PAM tal como o **teapop** pode usar bancos de dados (tal como o postgresql e o mysql) para autenticação do usuário.

FIXME: Verifique: uw-imapd também precisa ser configurado com autenticação do usuário através de PAM...

Recebendo mensagens de forma segura

A leitura/recebimento de mensagens é o protocolo de texto puro mais comum. Se usar ou POP3 ou IMAP para obter suas mensagens, você enviará sua senha em texto plano através da rede, assim praticamente qualquer um poderá ler suas mensagens de agora em diante. Ao invés disto, utiliza-se SSL (Secure Sockets Layer) para receber seus e-mails. A outra alternativa é utilizar o ssh, se tiver uma conta shell na máquina que atua como seu servidor POP ou IMAP. Aqui está um arquivo de configuração `fetchmailrc` básico para demonstrar isto:

```
poll my-imap-mailserver.org via "localhost"
  with proto IMAP port 1236
      user "ref" there with password "hackme" is alex here warnings 3600
    folders
      .Mail/debian
    preconnect 'ssh -f -P -C -L 1236:my-imap-mailserver.org:143 -l ref
     my-imap-mailserver.org sleep 15 </dev/null > /dev/null'
```

A linha preconnect é importante. Ela executa uma seção ssh e cria o túnel necessário, que automaticamente redireciona conexões para localhost da porta 1236 para o servidor de mensagens IMAP, mas de forma criptografada. Outra possibilidade será usar o fetchmail com características ssl.

Se deseja fornecer serviços de mensagens criptografadas como POP e IMAP,`apt-get install stunnel` e inicie seus daemons da seguinte forma:

```
stunnel -p /etc/ssl/certs/stunnel.pem -d pop3s -l /usr/sbin/popd
```

Este comando direciona as conexões do daemon fornecido (-l) para a porta (-d) e utiliza o certificado ssl especificado (-p).

Tornando o BIND mais seguro

Existem diferentes métodos que podem ser usados para deixar o daemon de serviços de Domínio mais seguro, que são parecidos com os mostrados considerados quando tornamos qualquer determinado serviço mais seguro:

- configurando o próprio daemon adequadamente assim ele não poderá ser abusado de fora (veja “Configuração do Bind para evitar má utilização”) Isto inclui limitar requisições de clientes: transferências de zonas e pesquisas recursivas.

- limitar o acesso do daemon ao próprio servidor assim se ele for usado para um corrompimento, a falha no sistema será limitada. Isto inclui executar o daemon como um usuário não-privilegiado (veja “Alterando o usuário do BIND”) e fazer ele rodar dentro um chroot (see “Executando o servidor de nomes em uma jaula chroot”)

Configuração do Bind para evitar má utilização

Você deverá restringir algumas das informações que são servidas pelo BIND para clientes externos, assim não poderão ser usadas para obter informações sobre sua empresa que não deseja dar. Isto inclui adicionar as seguintes opções: *allow-transfer*, *allow-query*, *allow-recursion* e *version*. Você pode ou limitar esta seção global (assim aplicando a todas as zonas que são servidas) ou por zona. Esta informação está incluída no pacote bind-doc, leia mais sobre isto em `/usr/share/doc/bind/html/index.html` assim que o pacote for instalado.

Imagine que seu servidor (um servidor básico contendo múltiplos endereços) está conectado à Internet e à sua rede interna (seu endereço IP é 192.168.1.2), você não vai querer oferecer qualquer serviço para os computadores. Você poderá restringir o bind incluindo o seguinte no `/etc/bind/named.conf`:

```
options {
\t    allow-query { 192.168.1/24; } ;
\t    allow-transfer { none; } ;
\t    allow-recursion { 192.168.1/24; } ;
\t    listen-on { 192.168.1.2; } ;
\t    forward { only; } ;
\t    forwarders { A.B.C.D; } ;
};
```

A opção *listen-on* faz o BIND ser executado somente na interface que tem o endereço interno, mas, até mesmo se esta interface for a mesma que te conecta a internet (caso estiver usando NAT, por exemplo), as requisições serão aceitas somente se estiverem vindo de suas máquinas internas. Se o sistema tiver múltiplas interfaces e a opção *listen-on* não estiver presente, somente usuários internos poderão fazer requisições, mas, como a porta está acessível para possíveis invasores externos, eles podem tentar travar (ou tentar realizar ataques de estouro de buffer) no servidor DNS. Você poderia até fazê-lo escutar somente em 127.0.0.1, se não estiver oferecendo o serviço de DNS em qualquer outro sistema além do seu.

O registro version.bind na classe chaos contém a versão do processo do bind atualmente em execução. Esta informação é freqüentemente usada por scaneadores automáticos e individualmente por pessoas maliciosas que desejam determinar se o bind é vulnerável a um ataque específico. Oferecendo informações falsas ou não fornecendo informações ao registro version.bind, diminui a probabilidade que o servidor seja atacado baseado na versão publicada. Para fornecer sua própria versão, use a diretiva *version* da seguinte forma:

```
 options { ... várias opções aqui ...
version "Não disponível."; };
```

A alteração do registro version.bind não oferece proteção atualmente contra ataques, mas pode ser considerado útil para a segurança.

Um arquivo simples de configuração `named.conf` pode ser o seguinte:

```
acl internal {
        127.0.0.1/32;           // localhost
        10.0.0.0/8;             // interna
        aa.bb.cc.dd;            // IP da eth0
```

```
};

acl friendly {
        ee.ff.gg.hh;            // DNS escravo
        aa.bb.cc.dd;            // IP da eth0
        127.0.0.1/32;           // localhost
        10.0.0.0/8;             // interna
};

options {
        directory "/var/cache/bind";
        allow-query { internal; };
        allow-recursion { internal; };
        allow-transfer { none; };
};
// A partir daqui, a zona mysite.bogus é
// basicamente uma versão não modificada do padrão do Debian
logging {
        category lame-servers { null; };
        category cname { null; };
};

zone "." {
        type hint;
        file "/etc/bind/db.root";
};

zone "localhost" {
        type master;
        file "/etc/bind/db.local";
};

zone "127.in-addr.arpa" {
        type master;
        file "/etc/bind/db.127";
};

zone "0.in-addr.arpa" {
        type master;
        file "/etc/bind/db.0";
};

zone "255.in-addr.arpa" {
        type master;
        file "/etc/bind/db.255";
};

// zones I added myself
zone "mysite.bogus" {
        type master;
        file "/etc/bind/named.mysite";
        allow-query { any; };
        allow-transfer { friendly; };
};
```

Por favor (novamente) verifique o Sistema de Tratamento de Falhas a respeito do bind, especificamente http://bugs.debian.org/94760. Sinta-se livre para contribuir para relatar falhas se achar que podem adicionar informações úteis.

Alterando o usuário do BIND

Com relação a limitação de privilégios do BIND, você deverá estar ciente que se um usuário não root executa o BIND, então o BIND não detectará novas interfaces automaticamente, por exemplo, se colocar uma placa PCMCIA no notebook. Verifique o arquivo README.Debian na documentação do named veja o diretório (`/usr/share/doc/bind/README.Debian`) para mais informações sobre este assunto. Ocorreram muitos problemas de segurança recentes relacionados com o BIND, assim a alteração do usuário é mais útil quando possível. Nós detalharemos os passos para fazer isto, no entanto, se quiser fazer isto de uma forma automática, tente o script fornecido em “Exemplo de script para alterar a instalação padrão do Bind.”.

Notice, in any case, that this only applies to BIND version 8. In the Debian packages for BIND version 9 (since the 9.2.1-5 version, available since *sarge*) the *bind* user is created and used by setting the OPTIONS variable in `/etc/default/bind9`. If you are using BIND version 9 and your name server daemon is not running as the *bind* user verify the settings on that file.

Para executar o BIND sob um usuário diferente, primeiro crie um usuário separado e um grupo (*não* é uma boa idéia usar o nobody ou nogroup para cada serviço que não estiver sendo executado como root). Neste exemplo, o usuário e grupo `named` serão usados. Você poderá fazer isto da seguinte forma:

```
addgroup named
adduser --system --home /home/named --no-create-home --ingroup named \
      --disabled-password --disabled-login named
```

Note que o usuário `named` será bastante restringido. Se você quiser, por alguma razão, ter uma configuração menos restrita, utilize:

```
adduser --system --ingroup named named
```

Agora, edite o arquivo `/etc/init.d/bind` com seu editor favorito e altere a linha que começa com

```
start-stop-daemon --start
```

para[4]

```
start-stop-daemon --start --quiet --exec /usr/sbin/named -- -g named -u named
```

Or you can change (create it if it does not exit) the default configuration file (`/etc/default/bind` for BIND version 8) and introduce the following:

```
OPTIONS="-u named -g named"
```

Altere as permissões dos arquivo que são usados pelo Bind, incluindo `/etc/bind/rndc.key`:

[4] Note que dependendo de sua versão do BIND você pode não ter a opção `-g`, mais precisamente se estiver usando a woody e instalando o bind9 (9.2.1-2.woody).

```
-rw-r-----    1 root     named          77 Jan  4 01:02 rndc.key
```

e onde o bind cria seu arquivo de pid, usando, por exemplo, `/var/run/named` ao invés de `/var/run`:

```
$ mkdir /var/run/named
$ chown named.named /var/run/named
$ vi /etc/named.conf
[ ... atualize o arquivo de configuração para sua nova localização ...]
options { ...
        pid-file "/var/run/named/named.pid";
};
[ ... ]
```

Also, in order to avoid running anything as root, change the `reload` line in the init.d script by substituting:

```
reload)
       /usr/sbin/ndc reload
```

to:

```
reload)
        $0 stop
        sleep 1
        $0 start
```

Nota: Dependendo de sua versão do Debian, você deverá também alterar a linha `restart`. Isto foi corrigido na versão do Bind do Debian `1:8.3.1-2`.

Tudo que precisa fazer agora é reiniciar o bind via '/etc/init.d/bind restart', e então procurar em seu syslog pelas seguintes duas linhas, como estas:

```
Sep  4 15:11:08 nexus named[13439]: group = named
Sep  4 15:11:08 nexus named[13439]: user = named
```

Voilá! Seu named agora *não é executado* como root. Se desejar ler mais informações sobre porque o BIND não pode ser executado por um usuário não-root em sistemas Debian, verifique o sistema de tratamento de falhas, especificamente http://bugs.debian.org/50013 e http://bugs.debian.org/132582, http://bugs.debian.org/53550, http://bugs.debian.org/52745, e http://bugs.debian.org/128129. Sinta-se livre para contribuir para os relatórios de falhas se achar que pode adicionar informações úteis.

Executando o servidor de nomes em uma jaula chroot

Para obter o máximo de segurança no BIND, agora construa uma jaula chroot (veja “Paranóia geral do chroot e suid”) em torno do seu daemon. Existe um método fácil de se fazer isto: a opção `-t` (veja a named(8) página de manual ou a página 100 do http://www.nominum.com/content/documents/bind9arm.pdf). Isto instruirá o Bind a fazer uma jaula de si mesmo em um diretório especificado sem a necessidade de configurar uma jaula chroot e se preocupar com as bibliotecas dinâmicas. Os únicos arquivos que precisam estar na jaula são:

```
dev/null
etc/bind/       - deverá ter o named.conf e todas as zonas do servidor
```

```
sbin/named-xfer - se fizer transferências de nomes
var/run/named/  - deverá ter a pid e o nome do servidor de cache (se tiver)
                  este diretório precisa ter permissões de gravação para o
                  usuário named.
var/log/named   - se configurar o log para um arquivo, este precisa ter permissões
                  de gravação para o usuário named
dev/log         - o syslogd deverá estar escutando aqui caso o named estiver
                  configurado para realizar logs através dele.
```

Para seu daemon do Bind funcionar adequadamente, ele precisará de permissões nos arquivos do named. Esta é uma tarefa simples, pois os arquivos de configuração estão sempre localizados em `/etc/named/`. Tenha em mente que ele somente precisa de acesso de leitura aos arquivos de zonas, a não ser que seja um DNS secundário ou servidor de cache de nomes. Se este é seu caso, você terá que dar permissões completas para as zonas necessárias (assim as zonas transferidas do servidor principal funcionarão).

Adicionalmente, mais detalhes sobre o Bind e chroot pode ser encontrados no http://www.tldp.org/HOWTO/Chroot-BIND-HOWTO.html (relacionado com o Bind 9) e http://www.tldp.org/HOWTO/Chroot-BIND8-HOWTO.html (relacionado com o Bind 8). Este mesmo documento deverá estar disponível através da instalação do doc-linux-text (versão texto) ou doc-linux-html (versão html). Outro documento útil é http://web.archive.org/web/20011024064030/http://www.psionic.com/papers/dns/dns-linux.

If you are setting up a full chroot jail (i.e. not just `-t`) for Bind in Debian, make sure you have the following files in it[5]:

```
dev/log - o syslogd deverá estar escutando aqui
dev/null
etc/bind/named.conf
etc/localtime
etc/group - com somente uma linha simples: "named:x:GID:"
etc/ld.so.cache - gerado com o ldconfig
lib/ld-2.1.3.so
lib/libc-2.1.3.so
lib/ld-linux.so.2 - link simbólico para ld-2.1.3.so
lib/libc.so.6 - link simbólico para libc-2.1.3.so
sbin/ldconfig - pode ser apagado após configurar a jaula chroot
sbin/named-xfer - se fizer transferências de nomes
var/run/
```

Também modifique o **syslogd** para escutar no $CHROOT/dev/log assim o servidor de nomes poderá gravar entradas do syslog no log local do sistema.

Se deseja evitar problemas com bibliotecas dinâmicas, você poderá compilar o binário estaticamente. Você poderá usar o **apt-get** para fazer isto, com a opção `source`. Ele pode até mesmo baixar os pacotes que precisa para compila-los adequadamente. Você deverá fazer algo similar a isto:

```
$ apt-get source bind
# apt-get build-dep bind
$ cd bind-8.2.5-2
  (edit src/port/linux/Makefile so CFLAGS includes the '-static'
   option)
$ dpkg-buildpackage -rfakeroot -uc -us
```

[5] This setup has not been tested for new release of Bind yet.

```
$ cd ..
# dpkg -i bind-8.2.5-2*deb
```

After installation, you will need to move around the files to the chroot jail[6] you can keep the `init.d` scripts in `/etc/init.d` so that the system will automatically start the name server, but edit them to add `--chroot /location_of_chroot` in the calls to **start-stop-daemon** in those scripts or use the *-t* option for BIND by setting it in the OPTIONS argument at the `/etc/default/bind` (for version 8) or `/etc/default/bind9` (for version 9) configuration file.

Para mais informações sobre como configurar jaulas chroot veja “Paranóia geral do chroot e suid”.

FIXME, merge info from http://people.debian.org/~pzn/howto/chroot-bind.sh.txt, http://www.cryptio.net/~ferlatte/config/ (Debian-specific), http://web.archive.org/web/20021216104548/http://www.psionic.com/papers/whitep01.html and http://csrc.nist.gov/fasp/FASPDocs/NISTSecuringDNS.htm.

Tornando o Apache mais seguro

FIXME: Adicionar conteúdo: os módulos fornecidos com a instalação padrão do Apache (sob /usr/lib/apache/X.X/mod_*) e módulos que podem ser instalados separadamente pelos pacotes libapache-mod-XXX.

Você poderá limitar o acesso ao servidor Apache se você somente deseja usar ele internamente (para propósitos de testes, para acessar os arquivos do doc-central, etc..) e não deseja que pessoas de fora o acessem. Para fazer isto, use as diretivas `Listen` ou `BindAddress` no `/etc/apache/http.conf`.

Using Listen:

```
Listen 127.0.0.1:80
```

Using BindAddress:

```
BindAddress 127.0.0.1
```

Então reinicie o apache com `/etc/init.d/apache restart` e você verá que ele somente esperará por requisições na interface loopback.

Em qualquer caso, se não estiver usando todas as funcionalidades fornecidas pelo Apache, você poderá querer dar uma olhada em outros servidores web fornecidos no Debian, como o dhttpd.

A http://httpd.apache.org/docs/misc/security_tips.html fornece informações relacionadas com medidas de segurança a serem tomadas no servidor web Apache (estes mesmos passos são oferecidos no Debian através do pacote apache-doc).

Mais informações sobre restrições do Apache configurando uma jaula chroot são mostradas em “Chroot environment for Apache”.

Proibindo a publicação de conteúdo dos usuários

A instalação padrão do Apache no Debian permite que usuários publiquem conteúdo sob o diretório `$HOME/public_html`. Este conteúdo pode ser pego remotamente usando uma URL tal como: http://your_apache_server/~user.

[6] Unless you use the `instdir` option when calling **dpkg** but then the chroot jail might be a little more complex.

Se não quiser permitir isto, você deverá alterar o arquivo de configuração `/etc/apache/http.conf` comentando a linha:

```
LoadModule userdir_module /usr/lib/apache/1.3/mod_userdir.so
```

If you are using Apache 2.0 you must remove the file `/etc/apache2/mods-enabled/userdir.load` or restrict the default configuration by modifying `/etc/apache2/mods-enabled/userdir.conf`.

However, if the module was linked statically (you can list the modules that are compiled in running `apache -l`) you must add the following to the Apache configuration file:

```
Userdir disabled
```

Um invasor ainda pode usar enumeração de usuário, pois a resposta do servidor será um *403 Permissão negada* e não um *404 Não disponível*.

Permissões de arquivos de log

Os arquivos de log do Apache, desde a 1.3.22-1, tem como dono o usuário 'root' e grupo 'adm' com permissões 640, estas permissões são alteradas após o rotacionamento de logs. Um intruso que acessou o sistema através do servidor web não será capaz (sem escalação de privilégios) de remover entradas antigas do log.

Arquivos da Web Publicados

Os arquivos do Apache estão localizados sob `/var/www`. Apenas após a instalação o arquivo de configuração padrão fornecerá algumas informações sobre o sistema (principalmente que é um sistema Debian executando o Apache). As páginas web padrões tem como dono o usuário root e grupo root por padrão, enquanto o processo do Apache é executado como o usuário e grupo www-data. Isto torna difícil para invasores que comprometem o sistema através do servidor web, desfigurarem o site. Você deverá, é claro, substituir as páginas padrões por suas próprias (que fornecem informações que não deseja mostrar para pessoas de fora).

Tornando o finger mais seguro

Se desejar executar o serviço finger, primeiro pergunte a você mesmo porque o deseja. Se precisar dele, você verá que o Debian fornece vários daemons de finger (saída do comando **apt-cache search fingerd**):

- cfingerd - Daemon de finger configurável
- efingerd - Outro daemon de finger para unix, capaz de ajustes finos em sua saída.
- ffingerd - um daemon seguro do finger
- fingerd - Servidor remoto de informações do usuário.
- xfingerd - BSD-like daemon de finger com suporte a qmail.

O ffingerd é o daemon de finger recomendado se estiver usando-o em serviços públicos. Em qualquer caso, você é encorajado, quando estiver configurando através do inetd, xinetd ou tcpserver, a: limitar o número de processos que podem ser executados ao mesmo tempo, limitando o acesso ao daemon de finger

de um número determinado de máquinas (usando o tcp wrappers) e escutando somente nas interfaces onde deve operar.

Paranóia geral do chroot e suid

O **chroot** é uma das mais poderosas possibilidades para restringir um daemon, ou um usuário ou outro serviço. Apenas imagine uma jaula em torno de seu alvo, onde o alvo não pode escapar dela (normalmente, mas existem várias condições que permitam que um escape de tal jaula). Se não confia em um usuário ou em um serviço, você poderá criar um ambiente root modificado para ele. Isto poderá usar algum espaço do disco para copiar todos os executáveis requeridos, assim como bibliotecas, na jaula. Mas então, até mesmo se o usuário fizer algo malicioso, o escopo do ano é limitado a jaula.

Muitos serviços executados como daemons poderão se beneficiar deste tipo de técnica. Os daemons que você instala no Debian não virão, no entanto, dentro de chroot [7] por padrão.

Isto inclui: servidores de nomes (tal como o **bind**), servidores web (tal como o **apache**), servidores de mensagens (tal como o **sendmail** e servidores ftp (tal como o **wu-ftpd**). Provavelmente basta dizer que a complexibilidade do BIND é a razão de que ele foi exposto a vários ataques nos últimos anos (see “Tornando o BIND mais seguro”).

No entanto, o Debian não oferece muitos programas que podem ajuda-lo a configurar um ambiente **chroot**. Veja “Criando automaticamente ambientes chroots”.

De qualquer maneira, se executar qualquer serviço em seu sistema, considere torná-lo mais seguro o possível. Isto inclui: revogar os privilégios de root, executá-lo em um ambiente seguro (tal como uma jaula chroot) ou substituí-lo por um equivalente mais seguro.

No entanto, já esteja avisado que uma jaula **chroot** pode ser quebrada se o usuário dentro dela for o superusuário. Assim você deverá estar certo que o serviço está sendo executado por um usuário não privilegiado. Limitando seu ambiente, estará limitando os arquivos lidos/executáveis que o serviço poderá acessar, assim, limitando as possibilidade de uma escalação privilegiada usar as vulnerabilidade de segurança locais do sistema. Até mesmo nesta situação, você não poderá ter certeza completa de que lá não existe métodos para um invasor inteligente quebrar a jaula. Usando somente programas de servidor que tem a reputação de serem seguidos é uma boa medida adicional. Até mesmo minúsculos furos como arquivos abertos podem serem usados por um invasor com conhecimentos para quebrar o sistema. Após tudo isto, o **chroot** não foi designado como uma ferramenta de segurança, mas como uma ferramenta de testes.

Criando automaticamente ambientes chroots

Existem diversos programas que fazem automaticamente o chroot de servidores e serviços. O Debian atualmente (aceita em maio de 2002) fornece o Wietse Venema's **chrootuid** no pacote chrootuid, assim como o pacote compartment e makejail. Estes programas podem criar um ambiente restritivo para a execução de qualquer programa (**chrootuid** lhe permite até executa-lo como um usuário restrito).

Algumas destas ferramentas podem ser usadas para criar facilmente um ambiente chroot. O programa **makejail** por exemplo, pode criar e atualizar uma jaula chroot com arquivos de configuração pequenos (ele fornece modelos de configuração para o **bind**, **apache**, **postgresql** e **mysql**). Ele tenta adivinhar e instalar na jaula todos os arquivos requeridos pelo daemon usando o **strace**, **stat** e dependências de pacotes do Debian. Mais informações podem ser obtidas em http://www.floc.net/makejail/. O **Jailer** é uma ferramenta similar que pode ser obtida de http://www.balabit.hu/downloads/jailer/ e também está disponível como um pacote do Debian GNU.

[7] Eles não tentarão ser executados sob *mínimo privilégio* que inclui a execução de daemons com seus próprios usuários ao invés de tê-los executando como root

Paranóia geral sobre senhas em texto puro

Você deverá tentar evitar qualquer serviço de rede que envia e receba senhas em texto puro através da rede, como o FTP/Telnet/NIS/RPC. O autor recomenda usar o ssh ao invés de telnet e ftp para qualquer um.

Tenha em mente que migrando do telnet para o ssh, mas continuando a usar outros protocolos de texto puro não aumenta sua segurança de qualquer modo! O melhor é remover o ftp, telnet, pop, imap, http e substituí-los por seus respectivos serviços criptografados. Você deverá considerar mover estes para suas versões SSL, ftp-ssl, telnet-ssl, pop-ssl, https ...

A maioria dos listados acima se aplicam para cada sistema Unix (você os encontrará se ler qualquer documento relacionado a tornar um sistema Linux (e outros tipos e Unix) mais seguro.

Desativando o NIS

Você não deverá usar o NIS, o Serviço de Informações de Rede, se possível, pois ele permite o compartilhamento de senha. Isto pode ser altamente inseguro se sua configuração for corrompida.

Se precisar de compartilhamento de senhas entre máquinas, você deverá considerar a adoção de outras alternativas. Por exemplo, a configuração de um servidor LDAP e o PAM para contactar o servidor LDAP para autenticação dos usuários. Você poderá encontrar uma configuração detalhada na http://www.tldp.org/HOWTO/LDAP-HOWTO.html (`/usr/share/doc/HOWTO/en-txt/LDAP-HOWTO.txt.gz`).

You can read more about NIS security in the http://www.tldp.org/HOWTO/NIS-HOWTO.html (`/usr/share/doc/HOWTO/en-txt/NIS-HOWTO.txt.gz`).

FIXME (jfs): Adicionar detalhes de como configurar isto no Debian

Tornando serviços RPC mais seguros

You should disable RPC if you do not need it.

Remote Procedure Call (RPC) is a protocol that programs can use to request services from other programs located on different computers. The **portmap** service controls RPC services by mapping RPC program numbers into DARPA protocol port numbers; it must be running in order to make RPC calls.

RPC-based services have had a bad record of security holes, although the portmapper itself hasn't (but still provides information to a remote attacker). Notice that some of the DDoS (distributed denial of service) attacks use RPC exploits to get into the system and act as a so called agent/handler.

You only need RPC if you are using an RPC-based service. The most common RPC-based services are NFS (Network File System) and NIS (Network Information System). See the previous section for more information about NIS. The File Alteration Monitor (FAM) provided by the package fam is also an RPC service, and thus depends on portmap.

Os serviços NFS são muito importante em algumas redes. Se este for o caso para você, então terá que encontrar um balanceamento de segurança e usabilidade para sua rede. (Você poderá ler mais sobre a segurança em NFS no http://www.tldp.org/HOWTO/NFS-HOWTO.html (`/usr/share/doc/HOWTO/en-txt/NFS-HOWTO.txt.gz`).)

Desativando completamente os serviços RPC

A desativação do portmap é bem simples. Existem diversos diferentes métodos. O mais simples no sistema Debian 3.0 e mais novos é desistalar o pacote portmap. Se estiver executando uma versão antiga do Debian,

terá que desativar o serviço como visto em “Desabilitando daemons de serviço”, porque o programa é parte do pacote net-base (que não pode ser removido sem quebrar o sistema).

Notice that some desktop environments (notably, GNOME) use RPC services and need the portmapper for some of the file management features. If this is your case, you can limit the access to RPC services as described below.

Limitando o acesso a serviços RPC

Infelizmente em alguns casos a remoção dos serviços RPC não é uma opção. Alguns serviços de desktop locais (notavelmente o fam da SGI) são baseados em RPC e assim precisam de um portmapper local. Isto significa que sob algumas situações, os usuários que estiverem instalando um ambiente de desktop (como o GNOME) instalarão também o portmapper.

Existem diversas formas de limitar o acesso ao portmapper e aos serviços de RPC:

- Bloqueando o acesso as portas usadas por estes serviços com um firewall local (veja “Adicionando capacidades de firewall”).
- Bloquear o acesso a estes serviços usando tcp wrappers, pois o portmapper (e alguns serviços RPC) são compilados com a `libwrap` (veja “Usando os tcpwrappers”. Isto significa que você poderá bloquear o acesso a eles através do `hosts.allow` e `hosts.deny` na configuração do tcp wrappers.
- Desde a versão 5-5, o pacote portmap pode ser configurado para somente realizar conexões na interface loopback. Para fazer isto, modifique o arquivo `/etc/default/portmap`, e descomente a seguinte linha: `#OPTIONS="-i 127.0.0.1"` e reinicie o portmapper. Isto é suficiente para permitir que serviços RPC locais funcionem enquanto ao mesmo tempo evite que sistemas remotos os acessem (no entanto, veja “Desativando assuntos relacionados a weak-end de máquinas”.

Adicionando capacidades de firewall

The Debian GNU/Linux operating system has the built-in capabilities provided by the Linux kernel. If you install a recent Debian release (default kernel installed is 2.6) you will have **iptables** (netfilter) firewalling available[8].

Fazendo um firewall no sistema local

Você poderá usar regras de firewall como uma forma de restringir o acesso a seu sistema local e, até mesmo, limitar comunicações feitas através dele. As regras de firewall também podem ser usadas para proteger processos que podem não estar corretamente configurados, *não* fornecendo serviços para algumas redes, endereços IP, etc...

No entanto, este passo é mostrado por último neste manual basicamente porque é *muito* melhor não depender solenemente das capacidades de firewall para proteger um dado sistema. A segurança em um sistema é feita através de camadas, o firewall deve ser a última a ser adicionada, uma vez que todos os serviços foram ajustados para serem mais seguros. Você pode facilmente imaginar uma configuração em que o administrador descuidadamente remove as regras de firewall por alguma razão (problemas com a configuração, descuido, erro humano ...), este sistema pode estar aberto para um ataque se não existir outro reforço no sistema para protege-lo.

[8] Available since the kernel version 2.4 (which was the default kernel in Debian 3.0). Previous kernel versions (2.2, available in even older Debian releases) used **ipchains**. The main difference between **ipchains** and **iptables** is that the latter is based on *stateful packet inspection* which provides for more secure (and easier to build) filtering configurations. Older (and now unsupported) Debian distributions using the 2.0 kernel series needed the appropriate kernel patch.

Por outro lado, tendo regras de firewall no sistema local também evita que coisas ruins aconteçam. Até mesmo se os serviços fornecidos estão configurados de forma segura, um firewall pode proteger de má configurações ou de serviços instalados recentemente que ainda não foram configurados adequadamente. Também, uma configuração forte evitará que cavalos de tróia *chamem a origem* de funcionarem a não ser que o código do firewall seja removido. Note que um intruso *não* precisa de acesso de superusuário para instalar um cavalo de tróia localmente que pode ser controlado remotamente (pois a escuta a porta é permitido caso não sejam portas privilegiadas e as capacidades não foram removidas).

Assim, uma configuração apropriada de firewall é aquela com a política padrão deny, que é:

- conexões de entrada são permitidas somente para serviços locais por máquinas permitidas.
- conexões de saída somente são permitidas para serviços usados pelo seu sistema (DNS, web browsing, pop, email....) [9]
- a regra forward bloqueia tudo (a não ser que esteja protegendo outros sistemas, veja abaixo).
- todas as outras conexões de entrada ou saída são negadas.

Usando um firewall para proteger outros sistemas

Um firewall também pode ser instalado no Debian para proteger, com regras de filtragem, o acesso a sistemas *através* dela, limitando sua exposição na Internet. O firewall pode ser configurado para evitar que sistemas de fora da rede local acesse serviços (portas) que não são públicas. Por exemplo, em um servidor de mensagens, somente a porta 25 (onde o serviço de e-mail foi definido) precisa ser acessada de fora. Um firewall pode ser configurado para, até mesmo se existem outros serviços disponibilizados publicamente, descartar qualquer pacote (isto é conhecido como *filtragem*) direcionado a máquina.

Você pode até mesmo configurar a máquina Debian GNU/Linux como uma firewall bridge, i.e. um firewall de filtragem completamente transparente para a rede que deixa de lado um endereço IP e assim não pode ser atacada diretamente. Dependendo do kernel que tiver instalado, você poderá precisar fazer a instalação do patch de bridge no firewall e então ir para a seção *802.1d Ethernet Bridging* quando estiver configurando o kernel e uma nova opção *netfilter (firewalling) support*. Veja “Configurando uma ponte firewall” para mais detalhes sobre como fazer isto em um sistema Debian GNU/Linux).

Setting up a firewall

The default Debian installation, unlike other Linux distributions, does not yet provide a way for the administrator to setup a firewall configuration throughout the default installation but you can install a number of firewall configuration packages (see “Usando pacotes de Firewall”).

É claro que a configuração do firewall é sempre dependente de sistema e rede. Um administrador deverá conhecer de antemão qual é a estrutura da rede e os sistemas que deseja proteger, os serviços que precisam ser acessados e se ou não outras considerações de rede (como NAT ou roteamento) devem ser levadas em conta. Seja cuidadoso quando configurar seu firewall, como Laurence J. Lane diz no pacote iptables:

As ferramentas podem ser facilmente mal utilizadas, causando uma enorme quantidade de peso na consciência e cortando o acesso a um sistema. Não é terrivelmente incomum para um administrador de sistemas remotos travar si próprio fora de um sistema centenas de milhares de milhas de distância. É também possível que alguém deixe ele próprio fora de um computador em que o teclado está sob seus dedos. Por favor, use com a devida precaução.

[9] De forma diferente de firewalls pessoais em outros sistemas operacionais, o Debian GNU/Linux (ainda) não fornece uma interface de geração de firewall que possa fazer regras de limitação por processo ou usuário. No entanto, o código do iptables pode fazer isto (veja o módulo owner na página de manual iptables(8))

Lembre-se disto: apenas a instalação do iptables (ou do antigo código de firewall) não oferece qualquer proteção, apenas fornece o programa. Para ter um firewall, você precisa *configurá-lo!*

If you do not have a clue on how to set up your firewall rules manually consult the *Packet Filtering HOWTO* and *NAT HOWTO* provided by iptables for offline reading at `/usr/share/doc/iptables/html/`.

If you do not know much about firewalling you should start by reading the http://www.tldp.org/HOWTO/Firewall-HOWTO.html, install the doc-linux-text package if you want to read it offline. If you want to ask questions or need help setting up a firewall you can use the debian-firewall mailing list, see http://lists.debian.org/debian-firewall. Also see “Conhecimento necessário” for more (general) pointers on firewalls. Another good iptables tutorial is http://iptables-tutorial.frozentux.net/iptables-tutorial.html.

Usando pacotes de Firewall

A configuração manual de um firewall pode ser complicada para o administrador novato (e muitas vezes para até mesmo o expert). No entanto, a comunidade de software livre tem criado um número de ferramentas que podem ser usadas para configurar facilmente um firewall local. Esteja avisado desde já que algumas destas ferramentas são orientadas somente para a proteção local (também chamadas de *firewall pessoal*) e algumas são mais versáteis e podem ser usadas para configurar regras complexas para proteger todas as redes.

Alguns softwares que podem ser usados para configurar regras de firewall em um sistema Debian são:

- For desktop systems:
  - firestarter, a GNOME application oriented towards end-users that includes a wizard useful to quickly setup firewall rules. The application includes a GUI to be able to monitor when a firewall rule blocks traffic.
  - guarddog, a KDE based firewall configuration package oriented both to novice and advanced users.
  - knetfilter, a KDE GUI to manage firewall and NAT rules for iptables (alternative/competitor to the guarddog tool although slightly oriented towards advanced users).
  - fireflier, an interactive tool to create iptables rules based on traffic seen on the system and applications. It has a server-client model so you have to install both the server (fireflier-server) and one of the available clients, with one client available for different desktop environments: fireflier-client-gtk (Gtk + client), fireflier-client-kde (KDE client) and fireflier-client-qt (QT client).
- For servers (headless) systems:
  - fwbuilder, an object oriented GUI which includes policy compilers for various firewall platforms including Linux' netfilter, BSD's pf (used in OpenBSD, NetBSD, FreeBSD and MacOS X) as well as router's access-lists. It is similar to enterprise firewall management software. Complete fwbuilder's functionality is also available from the command line.
  - shorewall, a firewall configuration tool which provides support for IPsec as well as limited support for traffic shaping as well as the definition of the firewall rules. Configuration is done through a simple set of files that are used to generate the iptables rules.
  - bastille, this hardening application is described in Capítulo 6, *Fortalecimento automático de sistemas Debian*. One of the hardening steps that the administrator can configure is a definition of the allowed and disallowed network traffic that is used to generate a set of firewall rules that the system will execute on startup.

Lots of other iptables frontends come with Debian; an extensive list comparing the different packages in Debian is maintained at the http://wiki.debian.org/Firewalls.

Notice that some of the packages outlined previously will introduce firewalling scripts to be run when the system boots. Test them extensively before rebooting or you might find yourself locked from the box. If you mix different firewalling packages you can have undesired effects, usually, the firewalling script that runs last will be the one that configures the system (which might not be what you intend). Consult the package documentation and use either one of these setups.

As mentioned before, some programs, like firestarter, guarddog and knetfilter, are administration GUIs using either GNOME or KDE (last two). These applications are much more user-oriented (i.e. for home users) than some of the other packages in the list which might be more administrator-oriented. Some of the programs mentioned before (like **bastille**) are focused at setting up firewall rules to protect the host they run in but are not necessarily designed to setup firewall rules for firewall hosts that protect a network (like **shorewall** or **fwbuilder**).

There is yet another type of firewall application: application proxies. If you are looking into setting up an enterprise-level firewall that does packet filtering and provides a number of transparent proxies that can do fine-grain traffic analysis you should consider using zorp, which provides this in a single program. You can also manually setup this type of firewall host using the proxies available in Debian for different services like for DNS using bind (properly configured), dnsmasq, pdnsd or totd for FTP using frox or ftp-proxy, for X11 using xfwp, for IMAP using imapproxy, for mail using smtpd, or for POP3 using p3scan. For other protocols you can either use a generic TCP proxy like simpleproxy or a generic SOCKS proxy like dante-server, tsocks or socks4-server. Typically, you will also use a web caching system (like squid) and a web filtering system (like squidguard or dansguardian).

Manual init.d configuration

Another possibility is to manually configure your firewall rules through an init.d script that will run all the **iptables** commands. Take the following steps:

- Review the script below and adapt it to your needs.
- Test the script and review the syslog messages to see which traffic is being dropped. If you are testing from the network you will want to either run the sample shell snippet to remove the firewall (if you don't type anything in 20 seconds) or you might want to comment out the *default deny* policy definitions (*-P INPUT DROP* and *-P OUTPUT DROP*) and check that the system will not drop any legitimate traffic.
- Move the script to `/etc/init.d/myfirewall`
- The below script takes advantage of Debian's use (since Squeeze) of dependency based boot sequencing. For more information see: https://wiki.debian.org/LSBInitScripts/DependencyBasedBoot and https://wiki.debian.org/LSBInitScripts. With the LSB headers set as they are in the script, insserv will automatically configure the system to start the firewall before any network is brought up, and stop the firewall after any network is brought down.

```
# insserv myfirewall
```

This is the sample firewall script:

```
#!/bin/sh
### BEGIN INIT INFO
# Provides:          myfirewall
# Required-Start:    $local_fs
# Required-Stop:     $local_fs
# Default-Start:     S
```

```
# Default-Stop:      0 6
# X-Start-Before:    $network
# X-Stop-After:      $network
# Short-Description: My custom firewall.
### END INIT INFO
#
# Simple example firewall configuration.
#
# Caveats:
# - This configuration applies to all network interfaces
#   if you want to restrict this to only a given interface use
#   '-i INTERFACE' in the iptables calls.
# - Remote access for TCP/UDP services is granted to any host,
#   you probably will want to restrict this using '--source'.
#
# chkconfig: 2345 9 91
# description: Activates/Deactivates the firewall at boot time
#
# You can test this script before applying with the following shell
# snippet, if you do not type anything in 10 seconds the firewall
# rules will be cleared.
#----------------------------------------------------------------
#  while true; do test=""; read  -t 20 -p "OK? " test ; \
#  [ -z "$test" ] && /etc/init.d/myfirewall clear ; done
#----------------------------------------------------------------

PATH=/bin:/sbin:/usr/bin:/usr/sbin

# Services that the system will offer to the network
TCP_SERVICES="22" # SSH only
UDP_SERVICES=""
# Services the system will use from the network
REMOTE_TCP_SERVICES="80" # web browsing
REMOTE_UDP_SERVICES="53" # DNS
# Network that will be used for remote mgmt
# (if undefined, no rules will be setup)
# NETWORK_MGMT=192.168.0.0/24
# If you want to setup a management network (i.e. you've uncommented
# the above line) you will need to define the SSH port as well (i.e.
# uncomment the below line.) Remember to remove the SSH port from the
# TCP_SERVICES string.
# SSH_PORT="22"

if ! [ -x /sbin/iptables ]; then
    exit 0
fi

fw_start () {

  # Input traffic:
  /sbin/iptables -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
  # Services
  if [ -n "$TCP_SERVICES" ] ; then
  for PORT in $TCP_SERVICES; do
```

```
  /sbin/iptables -A INPUT -p tcp --dport ${PORT} -j ACCEPT
done
fi
if [ -n "$UDP_SERVICES" ] ; then
for PORT in $UDP_SERVICES; do
  /sbin/iptables -A INPUT -p udp --dport ${PORT} -j ACCEPT
done
fi
# Remote management
if [ -n "$NETWORK_MGMT" ] ; then
  /sbin/iptables -A INPUT -p tcp --src ${NETWORK_MGMT} --dport ${SSH_PORT} -j AC
fi
# Remote testing
/sbin/iptables -A INPUT -p icmp -j ACCEPT
/sbin/iptables -A INPUT -i lo -j ACCEPT
/sbin/iptables -P INPUT DROP
/sbin/iptables -A INPUT -j LOG

# Output:
/sbin/iptables -A OUTPUT -j ACCEPT -o lo
/sbin/iptables -A OUTPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
# ICMP is permitted:
/sbin/iptables -A OUTPUT -p icmp -j ACCEPT
# So are security package updates:
# Note: You can hardcode the IP address here to prevent DNS spoofing
# and to setup the rules even if DNS does not work but then you
# will not "see" IP changes for this service:
/sbin/iptables -A OUTPUT -p tcp -d security.debian.org --dport 80 -j ACCEPT
# As well as the services we have defined:
if [ -n "$REMOTE_TCP_SERVICES" ] ; then
for PORT in $REMOTE_TCP_SERVICES; do
  /sbin/iptables -A OUTPUT -p tcp --dport ${PORT} -j ACCEPT
done
fi
if [ -n "$REMOTE_UDP_SERVICES" ] ; then
for PORT in $REMOTE_UDP_SERVICES; do
  /sbin/iptables -A OUTPUT -p udp --dport ${PORT} -j ACCEPT
done
fi
# All other connections are registered in syslog
/sbin/iptables -A OUTPUT -j LOG
/sbin/iptables -A OUTPUT -j REJECT
/sbin/iptables -P OUTPUT DROP
# Other network protections
# (some will only work with some kernel versions)
echo 1 > /proc/sys/net/ipv4/tcp_syncookies
echo 0 > /proc/sys/net/ipv4/ip_forward
echo 1 > /proc/sys/net/ipv4/icmp_echo_ignore_broadcasts
echo 1 > /proc/sys/net/ipv4/conf/all/log_martians
echo 1 > /proc/sys/net/ipv4/ip_always_defrag
echo 1 > /proc/sys/net/ipv4/icmp_ignore_bogus_error_responses
echo 1 > /proc/sys/net/ipv4/conf/all/rp_filter
echo 0 > /proc/sys/net/ipv4/conf/all/send_redirects
echo 0 > /proc/sys/net/ipv4/conf/all/accept_source_route
```

```
}

fw_stop () {
  /sbin/iptables -F
  /sbin/iptables -t nat -F
  /sbin/iptables -t mangle -F
  /sbin/iptables -P INPUT DROP
  /sbin/iptables -P FORWARD DROP
  /sbin/iptables -P OUTPUT ACCEPT
}

fw_clear () {
  /sbin/iptables -F
  /sbin/iptables -t nat -F
  /sbin/iptables -t mangle -F
  /sbin/iptables -P INPUT ACCEPT
  /sbin/iptables -P FORWARD ACCEPT
  /sbin/iptables -P OUTPUT ACCEPT
}


case "$1" in
  start|restart)
    echo -n "Starting firewall.."
    fw_stop
    fw_start
    echo "done."
    ;;
  stop)
    echo -n "Stopping firewall.."
    fw_stop
    echo "done."
    ;;
  clear)
    echo -n "Clearing firewall rules.."
    fw_clear
    echo "done."
    ;;
  *)
    echo "Usage: $0 {start|stop|restart|clear}"
    exit 1
    ;;
  esac
exit 0
```

Instead of including all of the iptables rules in the init.d script you can use the **iptables-restore** program to restore the rules saved using **iptables-save**. In order to do this you need to setup your rules, save the ruleset under a static location (such as `/etc/default/firewall`)

Configuring firewall rules through ifup

You can use also the network configuration in `/etc/network/interfaces` to setup your firewall rules. For this you will need to:

- Create your firewalling ruleset for when the interface is active.
- Save your ruleset with **iptables-save** to a file in `/etc`, for example `/etc/iptables.up.rules`
- Configure `/etc/network/interfaces` to use the configured ruleset:

```
iface eth0 inet static
        address x.x.x.x
        [.. interface configuration ..]
        pre-up iptables-restore < /etc/iptables.up.rules
```

You can optionally also setup a set of rules to be applied when the network interface is *down* creating a set of rules, saving it in `/etc/iptables.down.rules` and adding this directive to the interface configuration:

```
    post-down iptables-restore < /etc/iptables.down.rules
```

For more advanced firewall configuration scripts through ifupdown you can use the hooks available to each interface as in the `*.d/` directories called with **run-parts** (see run-parts(8) manual page).

Testing your firewall configuration

Testing your firewall configuration is as easy, and as dangerous, as just running your firewall script (or enabling the configuration you defined in your firewall configuration application). However, if you are not careful enough and you are configuring your firewall remotely (like through an SSH connection) you could lock yourself out.

There are several ways to prevent this. One is running a script in a separate terminal that will remove the firewall configuration if you don't feed it input. An example of this is:

```
$  while true; do test=""; read  -t 20 -p "OK? " test ; \
  [ -z "$test" ] && /etc/init.d/firewall clear ; done
```

Another one is to introduce a backdoor in your system through an alternate mechanism that allows you to either clear the firewall system or punch a hole in it if something goes awry. For this you can use knockd and configure it so that a certain port connection attempt sequence will clear the firewall (or add a temporary rule). Even though the packets will be dropped by the firewall, since **knockd** binds to the interface and *sees* you will be able to work around the problem.

Testing a firewall that is protecting an internal network is a different issue, you will want to look at some of the tools used for remote vulnerability assessment (see “Ferramentas de verificação remota de vulnerabilidades”) to probe the network from the outside in (or from any other direction) to test the effectiveness of the firewall configuation.

Capítulo 6. Fortalecimento automático de sistemas Debian

Após ler todas as informações dos capítulos anteriores você deve estar pensando "Eu tenho que fazer muitas coisas para ter meu sistema fortalecido, estas coisas não poderiam ser automatizadas?". A resposta é sim, mas tenha cuidado com ferramentas automatizadas. Algumas pessoas acreditam que uma ferramenta de fortalecimento não elimina a necessidade de uma boa administração. Assim não seja tolo em pensar que pode automatizar todo o processo e corrigir todos os problemas relacionados a ele. Segurança é um processo progressivo no qual o administrador deve estar participando e não somente ficar a espera deixando que as ferramentas façam todo o trabalho, já que nenhuma ferramenta poderia fazer: todas as implementações de políticas de segurança possíveis, cobrindo todos os ataques e todos os ambientes.

Since woody (Debian 3.0) there are two specific packages that are useful for security hardening. The harden package which takes an approach based on the package dependencies to quickly install valuable security packages and remove those with flaws, configuration of the packages must be done by the administrator. The bastille package that implements a given security policy on the local system based on previous configuration by the administrator (the building of the configuration can be a guided process done with simple yes/no questions).

Harden

The harden package tries to make it more easy to install and administer hosts that need good security. This package should be used by people that want some quick help to enhance the security of the system. It automatically installs some tools that should enhance security in some way: intrusion detection tools, security analysis tools, etc. Harden installs the following *virtual* packages (i.e. no contents, just dependencies or recommendations on others):

- harden-tools: tools to enhance system security (integrity checkers, intrusion detection, kernel patches...)
- harden-environment: helps configure a hardened environment (currently empty).
- harden-servers: removes servers considered insecure for some reason.
- harden-clients: removes clients considered insecure for some reason.
- harden-remoteaudit: tools to remotely audit a system.
- harden-nids: helps to install a network intrusion detection system.
- harden-surveillance: helps to install tools for monitoring of networks and services.

Useful packages which are not a dependence:

- harden-doc: provides this same manual and other security-related documentation packages.
- harden-development: development tools for creating more secure programs.

Be careful because if you have software you need (and which you do not wish to uninstall for some reason) and it conflicts with some of the packages above you might not be able to fully use harden. The harden packages do not (directly) do a thing. They do have, however, intentional package conflicts with known non-secure packages. This way, the Debian packaging system will not approve the installation of these packages. For example, when you try to install a telnet daemon with harden-servers, apt will say:

```
# apt-get install telnetd
The following packages will be REMOVED:
\tharden-servers
The following NEW packages will be installed:
telnetd
Do you want to continue (Y/n)
```

Isto deverá deixar o administrador mais tranqüilo, reconsiderando suas ações que serão tomadas.

Bastille Linux

http://bastille-linux.sourceforge.net/ is an automatic hardening tool originally oriented towards the Red Hat and Mandrake Linux distributions. However, the bastille package provided in Debian (since woody) is patched in order to provide the same functionality for Debian GNU/Linux systems.

O Bastille pode ser usado com diferentes interfaces com o usuário (todas são documentadas em sua própria página de manual no pacote da Debian) que permite o administrador a:

- Answer questions step by step regarding the desired security of your system (using InteractuveBastille(8)
- Usar a configuração padrão de segurança (entre três: Fraca, Moderada, e Paranóica) em um determinado sistema (servidor e estação de trabalho) e deixar o Bastille decidir que política de segurança que será implementada (usando BastilleChooser(8))
- Take a predefined configuration file (could be provided by Bastille or made by the administrator) and implement a given security policy (using AutomatedBastille(8)).

Capítulo 7. Infraestrutura do Debian Security

O time Debian Security

O Debian tem um Security Team (Time de Segurança), composto por cinco membros e duas secretárias que manipulam a segurança na distribuição *stable* (estável). Manipular a segurança significa que eles acompanham as vulnerabilidades que aparecem nos software (vendo foruns como bugtraq o vuln-dev) e determinam se a distribuição *stable* é afetada por eles.

Also, the Debian Security Team is the contact point for problems that are coordinated by upstream developers or organizations such as http://www.cert.org which might affect multiple vendors. That is, when problems are not Debian-specific. The contact point of the Security Team is mailto:team@security.debian.org which only the members of the security team read.

Informações sensíveis devem ser enviadas para o primeiro email e, em alguns casos, deve ser encriptada com a Debian Security Contact key (key ID 363CCD95).

Quando um provável problema for recebido pelo Security Team, ele investigará se a distribuição *stable* foi afetada e, caso positivo, uma correção será feita no código fonte base. Esta correção algumas vezes incluirá algum patch (que normalmente é mais recente que a versão distribuída pelo Debian). Após o teste da correção, novos pacotes são preparados e publicados em security.debian.org e podem ser baixados com o **apt** (veja "Executar uma atualização de segurança"). Ao mesmo tempo um *Debian Security Advisory* (DSA) é publicado no web site e enviado para a listas de email incluindo lists.debian.org/debian-security-announce e bugtraq.

Outras perguntas frequentes do Debian Security Team podems er encontradas em "Questões relacionadas ao time de segurança da Debian".

Debian Security Advisories

Debian Security Advisories são avisos emitidos quandos uma vulnerabilidade de segurança que afeta um pacote Debian é descoberta. Estes avisos, assinados por um membro do Security Team, inclui informação das versões afetadas assim como a localização das atualizações e seus MD5sums. Esta informação consiste de:

- número da versão para correção.
- tipo de problema.
- se ele é remoto ou localmente explorável.
- pequena descriçãod do pacote.
- descrição do problema.
- descrição da exploração.
- descrição da correção.

DSAs are published both on http://www.debian.org/ and in the http://www.debian.org/security/. Usually this does not happen until the website is rebuilt (every four hours) so they might not be present immediately. The preferred channel is the debian-security-announce mailing list.

Usuários interessados podem, porém, usar o canal RDF para baixar automaticamente as DSAs para seu computador. Algumas aplicações, como o **Evolution** (um cliente de email e assistente de informações pessoais) e o **Multiticker** (um applet do GNOME), podem ser usados para baixar os avisos automaticamente. O canal RDF está disponível em http://www.debian.org/security/dsa.rdf.

Os DSAs publicados no website podem ser atualizados após enviados para as listas de email. Uma atualização comum é adicionada através de referências ao banco de dados de vulnerabilidades de segurança. Além disso, traduções [1] dos DSAs não são enviadas para as listas de email mas são diretamente incluídas no site.

Referências sobre vulnerabilidades

Debian fornece uma referência completa em http://www.debian.org/security/crossreferences incluindo todas as remomendações publicadas desde 1998. Esta tabela é fornecida em complemento a http://cve.mitre.org/cve/refs/refmap/source-DEBIAN.html.

Você notará que esta tabela fornece referências aos bancos de dados como http://www.securityfocus.com/bid, http://www.cert.org/advisories/ and http://www.kb.cert.org/vuls assim como aos nomes CVE (veja abaixo). Estas referências são fornecidas para uso, porém apenas referências CVE são periodicamente revisadas e incluídas. Este recurso foi adicionado ao website em junho de 2002.

Uma das vantagens de adicionar referências ao banco de dados de vulnerabilidades é que:

- torna fácil aos usuários Debian ver e tratar com recomendações publicadas que já tenham sido resolvidas pelo Debian.
- administradores de sistema podem aprender mais sobre vulnerabilidades e seu impacto através das referências.
- esta informação pode ser usada para a checagem de vulnerabilidades referentes ao CVE e detectar avisos falsos. (veja “O scanner de vulnerabilidade X diz que meu sistema Debian é vulnerável!”).

Compatibilidade CVE

As recomendações de segurança, Debian Security Advisories eram http://www.debian.org/security/CVE-certificate.jpg[2] em fevereiro de 2004.

Debian developers understand the need to provide accurate and up to date information of the security status of the Debian distribution, allowing users to manage the risk associated with new security vulnerabilities. CVE enables us to provide standardized references that allow users to develop a https://cve.mitre.org/compatible/enterprise.html.

O projeto http://cve.mitre.org é mantido pela MITRE Corporation e fornece uma lista de nomes padronizados para vulnerabilidades e exposições de segurança.

Debian acredita que fornecer aos usuários informações relacionadas a segurança que afetem a distribuição é extremamente importante. A inclusão dos nomes CVE em avisos ajudam os usuários a associar vulnerabilidades genéricas com atualizações específicas, com redução do tempo gasto para manusear as vulnerabilidades. Além disso, é fácil o gerenciamento da segurança em um ambiente onde já existem ferramentas que utilizam o CVE, como redes ou sistemas de detecção de invasão, ou ferramentas de avalição de vulnerabilidades, mesmo que elas não sejam baseadas em uma distribuição Debian.

[1] Traduções estão disponíveis em dez idiomas
[2] O completo http://cve.mitre.org/compatible/phase2/SPI_Debian.html estava disponível para o CVE

Debian provides CVE names for all DSAs released since September 1998. All of the advisories can be retrieved on the Debian web site, and announcements related to new vulnerabilities include CVE names if available at the time of their release. Advisories associated with a given CVE name can be searched directly through the Debian Security Tracker (see below).

Em alguns casos você pode nãO encontrar um CVE em avisos publicados porque:

- No Debian os produtos não são afetados pela vulnerabilidades.
- Ainda não existe uma viso abordando a vulnerabilidade (ele pode ter sido informado para a http://bugs.debian.org/cgi-bin/pkgreport.cgi?tag=security mas uma correção ainda não ter sido testada a atualizada)
- Um aviso foi publicado antes que um CVE fosse assinado para a vulnerabilidade em questão (procure por uma atualização no web site)

Security Tracker

The central database of what the Debian security teams know about vulnerabilities is the http://security-tracker.debian.org. It cross references packages, vulnerable and fixed versions for different suites, CVE names, Debian bug numbers, DSA's and miscellaneous notes. It can be searched, e.g. by CVE name to see which Debian packages are affected or fixed, or by package to show unresolved security issues. The only information missing from the tracker is confidential information that the security team received under embargo.

The package **debsecan** uses the information in the tracker to report to the administrator of a system which of the installed packages are vulnerable, and for which updates are available to fix security issues.

Infraestrutura da segurança Debian

Uma vez que o Debian é normalmente suportado em um grande número de arquiteturas, administradores algumas ficam admirados se uma dada arquitetura levar mais tempo para receber atualizações de segurança. De fato, exceto em raras circunstâncias, atualizações estão disponíveis para todas as arquiteturas ao mesmo tempo.

Packages in the security archive are autobuilt, just like the regular archive. However, security updates are a little more different than normal uploads sent by package maintainers since, in some cases, before being published they need to wait until they can be tested further, an advisory written, or need to wait for a week or more to avoid publicizing the flaw until all vendors have had a reasonable chance to fix it.

Thus, the security upload archive works with the following procedure:

- Alguém encontra um problema de segurança.
- Alguém corrige o problema e atualiza security.debian.org (este *alguém* normalmente é um membro do Time de Segurança mas pode ser também um mantenedor de pacote com uma correção apropriada que contactou o time de segurança previamente). O Changelog inclui uma indicação *testing-security* ou *stable-security*.
- Ocorre o upload checado e processado por um sistema Debian e movido para queue/accepted, e buildds são notificados. Arquivos aqui podem ser acessados pelo time de segurança e (indiretamente) pelos buildds.

- O Security-enable buildds pega o pacote fonte (que tem prioridade sobre os builds normais), o constrói, e envia logs para o time de segurança.
- O time de segurança reproduz os logs, e novos pacotes construídos são enviados para queue/unchecked, onde são processados pelo sistema Debian, e movidos para queue/accepted.
- Quando o time de segurança verifica que o pacote fonte está aceitável (isto é, ele foi corretamente construído para todas as arquiteturas, corrigiu os problemas de segurança e não introduziu novos problemas) eles rodam um script que:
  - instala o pacote em um arquivo de segurança.
  - updates the `Packages`, `Sources` and `Release` files of security.debian.org in the usual way (**dpkg-scanpackages**, **dpkg-scansources**, ...).
  - configura um aviso modelo que o time de seguranaça pode encerrar os trabalhos.
  - (opcionalmente) envia os pacotes para as atualizações adequadas e eles podem ser incluídos assim que for possível.

Este procedimento, antes feito a mão, foi testado e usado completamente durante o estágio freeze do Debian 3.0 Woody (Julho de 2002). Graças a esta infraestrutura do Security Team foi possível ter pacotes atualizados prontos para o apache e OpenSSH para todas as arquiteturas suportadas (quase vinte) em menos de um dia.

Guia dos desenvolvedores de atualizações de seguranaça

Debian developers that need to coordinate with the security team on fixing in issue in their packages, can refer to the Developer's Reference section http://www.debian.org/doc/manuals/developers-reference/pkgs.html#bug-security.

Assinatura de pacote no Debian

Esta seção também pode ser chamada "como atualizar seu sistema Debian GNU/Linux em segurança" e mereçe sua própria seção basicamente porque é uma parte importante da infraestrutura de segurança. Assinatura de pacote é uma coisa importante porque evita alterações de pacotes distribuídos em mirros. Atualização automatica de software é um recurso importante mas também é importante remover ameaças de segurança que poderiam ajudar a distribuir cavalos de tróia e comprometer os sistemas durante as atualizações. [3].

FIXME: probably the Internet Explorer vulnerability handling. certificate chains has an impact on security updates on Microsoft Windows.

Debian does not provide signed packages but provides a mechanism available since Debian 4.0 (codename *etch*) to check for downloaded package's integrity[4]. For more information, see “Secure apt”.

This issue is better described in the http://www.cryptnet.net/fdp/crypto/strong_distro.html by V. Alex Brennen.

[3] Alguns sistemas operacionais já tiveram problemas com atualizações automáticas como http://www.cunap.com/~hardingr/projects/osx/exploit.html.

FIXME: probably the Internet Explorer vulnerability handling certificate chains has an impact on security updates on Microsoft Windows.

[4] Older releases, such as Debian 3.1 *sarge* can use this feature by using backported versions of this package management tool

O esquema proposto para checagem de assinatura dos pacotes

O esquema atual para checagem da assinatura dos pacotes usando **apt** é:

- o arquivo lançado incluirá o md5sum do Packages.gz (que contém os md5sum dos pacotes) e será assinado. A assinatura é de uma fonte certificada.
- A arquivo assinado é baixado pelo 'apt-get update' a armazenado com o Packages.gz.
- Quando o pacote está sendo instalado, ele primeiro é baixado, então o md5sum é gerado.
- A assinatura é checada (assinatura ok) e extraído o md5sum do arquivo Pakages.gz, este por sua vez é gerado e (se ok) o md5sum do pacote baixado é extraído.
- Se o md5sum do pacote baixado é o mesmo que o do Packages.gz, o pacote será instalado. Caso contrário o administrador será alertado e o pacote será colocado num cache (e o administrador pode decidir se instalará o pacote ou não). Se o pacote não estiver no Packages.gz e o administrador tiver configurado o sistema para só instalar pacotes checados, o pacote não será instalado.

A sequência seguinte de checagens MD5 do **apt** é capaz de verificar se o pacote origina de um release específico. Isto é menos flexível que a assinatura de cada pacote, mas pode ser combinada com este esquema também (veja abaixo).

This scheme is http://lists.debian.org/debian-devel/2003/12/msg01986.html in apt 0.6 and is available since the Debian 4.0 release. For more information see “Secure apt”. Packages that provide a front-end to apt need to be modified to adapt to this new feature; this is the case of **aptitude** which was http://lists.debian.org/debian-devel/2005/03/msg02641.html to adapt to this scheme. Front-ends currently known to work properly with this feature include **aptitude** and **synaptic**.

Assinatura de pacotes foi discutido no Debian por um bom tempo, para mais informações leia: http://www.debian.org/News/weekly/2001/8/ e http://www.debian.org/News/weekly/2000/11/.

Secure apt

The apt 0.6 release, available since Debian 4.0 *etch* and later releases, includes *apt-secure* (also known as *secure apt*) which is a tool that will allow a system administrator to test the integrity of the packages downloaded through the above scheme. This release includes the tool **apt-key** for adding new keys to apt's keyring, which by default includes only the current Debian archive signing key.

These changes are based on the patch for **apt** (available in http://bugs.debian.org/cgi-bin/bugreport.cgi?bug=203741) which provides this implementation.

Secure apt works by checking the distribution through the `Release` file, as discussed in “Per distribution release check”. Typically, this process will be transparent to the administrator although you will need to intervene every year[5] to add the new archive key when it is rotated, for more information on the steps an administrator needs to take a look at “Safely adding a key”.

This feature is still under development, if you believe you find bugs in it, please, make first sure you are using the latest version (as this package might change quite a bit before it is finally released) and, if running the latest version, submit a bug against the apt package.

[5] Until an automatic mechanism is developed.

You can find more information at http://wiki.debian.org/SecureApt and the official documentation: http://www.enyo.de/fw/software/apt-secure/ and https://web.archive.org/web/20070206063141/http://www.syntaxpolice.org/apt-secure/.

Per distribution release check

This section describes how the distribution release check mechanism works, it was written by Joey Hess and is also available at the http://wiki.debian.org/SecureApt.

Basic concepts

Here are a few basic concepts that you'll need to understand for the rest of this section.

A checksum is a method of taking a file and boiling it down to a reasonably short number that uniquely identifies the content of the file. This is a lot harder to do well than it might seem, and the most commonly used type of checksum, the MD5 sum, is in the process of being broken.

Public key cryptography is based on pairs of keys, a public key and a private key. The public key is given out to the world; the private key must be kept a secret. Anyone possessing the public key can encrypt a message so that it can only be read by someone possessing the private key. It's also possible to use a private key to sign a file, not encrypt it. If a private key is used to sign a file, then anyone who has the public key can check that the file was signed by that key. No one who doesn't have the private key can forge such a signature.

These keys are quite long numbers (1024 to 2048 digits or longer), and to make them easier to work with they have a key id, which is a shorter, 8 or 16 digit number that can be used to refer to them.

gpg is the tool used in secure apt to sign files and check their signatures.

apt-key is a program that is used to manage a keyring of gpg keys for secure apt. The keyring is kept in the file `/etc/apt/trusted.gpg` (not to be confused with the related but not very interesting `/etc/apt/trustdb.gpg`). **apt-key** can be used to show the keys in the keyring, and to add or remove a key.

`Release` checksums

A Debian archive contains a `Release` file, which is updated each time any of the packages in the archive change. Among other things, the `Release` file contains some MD5 sums of other files in the archive. An excerpt of an example `Release` file:

```
MD5Sum:
 6b05b392f792ba5a436d590c129de21f            3453 Packages
 1356479a23edda7a69f24eb8d6f4a14b            1131 Packages.gz
 2a5167881adc9ad1a8864f281b1eb959            1715 Sources
 88de3533bf6e054d1799f8e49b6aed8b             658 Sources.gz
```

The `Release` files also include SHA-1 checksums, which will be useful once MD5 sums become fully broken, however apt doesn't use them yet.

Now if we look inside a `Packages` file, we'll find more MD5 sums, one for each package listed in it. For example:

```
Package: uqm
Priority: optional
...
```

```
Filename: unstable/uqm_0.4.0-1_i386.deb
Size: 580558
MD5sum: 864ec6157c1eea88acfef44d0f34d219
```

These two checksums can be used to verify that you have downloaded a correct copy of the `Packages` file, with a md5sum that matches the one in the `Release` file. And when it downloads an individual package, it can also check its md5sum against the content of the `Packages` file. If apt fails at either of these steps, it will abort.

None of this is new in secure apt, but it does provide the foundation. Notice that so far there is one file that apt doesn't have a way to check: The Release file. Secure apt is all about making apt verify the `Release` file before it does anything else with it, and plugging this hole, so that there is a chain of verification from the package that you are going to install all the way back to the provider of the package.

Verification of the `Release` file

To verify the `Release` file, a gpg signature is added for the `Release` file. This is put in a file named `Release.gpg` that is shipped alongside the `Release` file. It looks something like this [6] , although only gpg actually looks at its contents normally:

```
-----BEGIN PGP SIGNATURE-----
Version: GnuPG v1.4.1 (GNU/Linux)

iD8DBQBCqKO1nukh8wJbxY8RAsfHAJ9hu8oGNRAl2MSmP5+z2RZb6FJ8kACfWvEx
UBGPVc7jbHHsg78EhMBlV/U=
=x6og
-----END PGP SIGNATURE-----
```

Check of `Release.gpg` by apt

Secure apt always downloads `Release.gpg` files when it's downloading `Release` files, and if it cannot download the `Release.gpg`, or if the signature is bad, it will complain, and will make note that the `Packages` files that the `Release` file points to, and all the packages listed therein, are from an untrusted source. Here's how it looks during an **apt-get update**:

```
W: GPG error: http://ftp.us.debian.org testing Release: The following signatures
 couldn't be verified because the public key is not available: NO_PUBKEY 010908312
```

Note that the second half of the long number is the key id of the key that apt doesn't know about, in this case that's 2D230C5F.

If you ignore that warning and try to install a package later, apt will warn again:

```
WARNING: The following packages cannot be authenticated!
  libglib-perl libgtk2-perl
Install these packages without verification [y/N]?
```

If you say Y here you have no way to know if the file you're getting is the package you're supposed to install, or if it's something else entirely that somebody that can intercept the communication against the server[7] has arranged for you, containing a nasty suprise.

[6] Technically speaking, this is an ASCII-armored detached gpg signature.
[7] Or has poisoned your DNS, or is spoofing the server, or has replaced the file in the mirror you are using, etc.

Note that you can disable these checks by running apt with --allow-unauthenticated.

It's also worth noting that newer versions of the Debian installer use the same signed `Release` file mechanism during their debootstrap of the Debian base system, before apt is available, and that the installer even uses this system to verify pieces of itself that it downloads from the net. Also, Debian does not currently sign the `Release` files on its CDs; apt can be configured to always trust packages from CDs so this is not a large problem.

How to tell apt what to trust

So the security of the whole system depends on there being a `Release.gpg` file, which signs a `Release` file, and of **apt** checking that signature using gpg. To check the signature, it has to know the public key of the person who signed the file. These keys are kept in apt's own keyring (`/etc/apt/trusted.gpg`), and managing the keys is where secure apt comes in.

By default, Debian systems come preconfigured with the Debian archive key in the keyring.

```
# apt-key list
/etc/apt/trusted.gpg
--------------------
pub   1024D/4F368D5D 2005-01-31 [expires: 2006-01-31]
uid                  Debian Archive Automatic Signing Key (2005) <ftpmaster@debian
```

Here 4F368D5D is the key id, and notice that this key was only valid for a one year period. Debian rotates these keys as a last line of defense against some sort of security breach breaking a key.

That will make **apt** trust the official Debian archive, but if you add some other apt repository to `/etc/apt/sources.list`, you'll also have to give **apt** its key if you want apt to trust it. Once you have the key and have verified it, it's a simple matter of running **apt-key add** *`file`* to add it. Getting the key and verifying it are the trickier parts.

Finding the key for a repository

The debian-archive-keyring package is used to distribute keys to **apt**. Upgrades to this package can add (or remove) gpg keys for the main Debian archive.

For other archives, there is not yet a standard location where you can find the key for a given apt repository. There's a rough standard of putting the key up on the web page for the repository or as a file in the repository itself, but no real standard, so you might have to hunt for it.

The Debian archive signing key is available at https://ftp-master.debian.org/keys.html.[8]

gpg itself has a standard way to distribute keys, using a keyserver that gpg can download a key from and add it to its keyring. For example:

```
$ gpg --keyserver pgpkeys.mit.edu --recv-key 2D230C5F
gpg: requesting key 2D230C5F from hkp server pgpkeys.mit.edu
gpg: key 2D230C5F: public key "Debian Archive Automatic Signing Key (2006) <ftpm
aster@debian.org>" imported
gpg: Total number processed: 1
gpg:               imported: 1
```

[8] "ziyi" is the name of the tool used for signing on the Debian servers, the name is based on the name of a http://en.wikipedia.org/wiki/Zhang_Ziyi.

You can then export that key from your own keyring and feed it to **apt-key**:

```
$ gpg -a --export 2D230C5F | sudo apt-key add -
gpg: no ultimately trusted keys found
OK
```

The "gpg: no ultimately trusted keys found" warning means that gpg was not configured to ultimately trust a specific key. Trust settings are part of OpenPGPs Web-of-Trust which does not apply here. So there is no problem with this warning. In typical setups the user's own key is ultimately trusted.

Safely adding a key

By adding a key to apt's keyring, you're telling apt to trust everything signed by the key, and this lets you know for sure that apt won't install anything not signed by the person who possesses the private key. But if you're sufficiently paranoid, you can see that this just pushes things up a level, now instead of having to worry if a package, or a `Release` file is valid, you can worry about whether you've actually gotten the right key. Is the key file from https://ftp-master.debian.org/keys.html mentioned above really Debian's archive signing key, or has it been modified (or this document lies).

It's good to be paranoid in security, but verifying things from here is harder. **gpg** has the concept of a chain of trust, which can start at someone you're sure of, who signs someone's key, who signs some other key, etc., until you get to the archive key. If you're sufficiently paranoid you'll want to check that your archive key is signed by a key that you can trust, with a trust chain that goes back to someone you know personally. If you want to do this, visit a Debian conference or perhaps a local LUG for a key signing [9].

If you can't afford this level of paranoia, do whatever feels appropriate to you when adding a new apt source and a new key. Maybe you'll want to mail the person providing the key and verify it, or maybe you're willing to take your chances with downloading it and assuming you got the real thing. The important thing is that by reducing the problem to what archive keys to trust, secure apt lets you be as careful and secure as it suits you to be.

Verifying key integrity

You can verify the fingerprint as well as the signatures on the key. Retrieving the fingerprint can be done for multiple sources, you can talk to Debian Developers on IRC, read the mailing list where the key change will be announced or any other additional means to verify the fingerprint. For example you can do this:

```
$ GET http://ftp-master.debian.org/ziyi_key_2006.asc | gpg --import
gpg: key 2D230C5F: public key "Debian Archive Automatic Signing Key (2006)
  <ftpmaster&debian.org>" imported
gpg: Total number processed: 1
gpg:               imported: 1
$ gpg --check-sigs --fingerprint 2D230C5F
pub   1024D/2D230C5F 2006-01-03 [expires: 2007-02-07]
      Key fingerprint = 0847 50FC 01A6 D388 A643  D869 0109 0831 2D23 0C5F
uid   Debian Archive Automatic Signing Key (2006) <ftpmaster@debian.org>
sig!3        2D230C5F 2006-01-03  Debian Archive Automatic Signing Key
                                  (2006) <ftpmaster@debian.org>
sig!         2A4E3EAA 2006-01-03  Anthony Towns <aj@azure.humbug.org.au>
```

[9] Not all apt repository keys are signed at all by another key. Maybe the person setting up the repository doesn't have another key, or maybe they don't feel comfortable signing such a role key with their main key. For information on setting up a key for a repository see “Release check of non Debian sources”.

```
sig!         4F368D5D 2006-01-03  Debian Archive Automatic Signing Key
                                  (2005) <ftpmaster@debian.org>
sig!         29982E5A 2006-01-04  Steve Langasek <vorlon@dodds.net>
sig!         FD6645AB 2006-01-04  Ryan Murray <rmurray@cyberhqz.com>
sig!         AB2A91F5 2006-01-04  James Troup <james@nocrew.org>
```

and then as in “Assinatura de pacote no Debian” check the trust path from your key (or a key you trust) to at least one of the keys used to sign the archive key. If you are sufficiently paranoid you will tell apt to trust the key only if you find an acceptable path:

```
$ gpg --export -a 2D230C5F | sudo apt-key add -
Ok
```

Note that the key is signed with the previous archive key, so theoretically you can just build on your previous trust.

Debian archive key yearly rotation

As mentioned above, the Debian archive signing key is changed each year, in January. Since secure apt is young, we don't have a great deal of experience with changing the key and there are still rough spots.

In January 2006, a new key for 2006 was made and the `Release` file began to be signed by it, but to try to avoid breaking systems that had the old 2005 key, the `Release` file was signed by that as well. The intent was that apt would accept one signature or the other depending on the key it had, but apt turned out to be buggy and refused to trust the file unless it had both keys and was able to check both signatures. This was fixed in apt version 0.6.43.1. There was also confusion about how the key was distributed to users who already had systems using secure apt; initially it was uploaded to the web site with no announcement and no real way to verify it and users were forced to download it by hand.

In January 2006, a new key for 2006 was made and the Release file began to be signed by it, but to try to avoid breaking systems that had the old 2005 key, the `Release` file was signed by that as well. In order to prevent confusion on the best distribution mechanism for users who already have systems using secure apt, the debian-archive-keyring package was introduced, which manages apt keyring updates.

Known release checking problems

One not so obvious problem is that if your clock is very far off, secure apt will not work. If it's set to a date in the past, such as 1999, apt will fail with an unhelpful message such as this:

```
W: GPG error: http://archive.progeny.com sid Release: Unknown error executing gpg
```

Although **apt-key** list will make the problem plain:

```
gpg: key 2D230C5F was created 192324901 seconds in the future (time warp or clock
gpg: key 2D230C5F was created 192324901 seconds in the future (time warp or clock
pub   1024D/2D230C5F 2006-01-03
uid                  Debian Archive Automatic Signing Key (2006) <ftpmaster@debian
```

If it's set to a date too far in the future, apt will treat the keys as expired.

Another problem you may encouter if using testing or unstable is that if you have not run **apt-get update** lately and **apt-get install** a package, apt might complain that it cannot be authenticated (why does it do this?). **apt-get update** will fix this.

Manual per distribution release check

Caso você queira adicionar os novos recursos de checagem de segurança e não queira rodar a versão experimental do apt (embora nõs realmente apreciemos o teste dele) você pode usar o script abaixo, fornecido por Anthony Towns. Este script pode automaticamente fazer algumas novas checagens de segurança para permitir ao usuário certificar-se que o software que está baixando corresponde aquele distribuído pelo Debian. Isto é para desenvolvedores Debian usarem em sistemas sem a funcionalidade de uploading dos sistemas tradicionais, ou mirrors que tem quase tudo mas não como o Debian, ou mirrors que fornecem dados da versão unstable sem conhecimento dos problemas de segurança.

Este código, renomeado como **apt-check-sigs**, deve ser usado da seguinte maneira:

```
# apt-get update
# apt-check-sigs
(...resultados...)
# apt-get dist-upgrade
```

Primeiro você precisa:

- get the keys the archive software uses to sign `Release` files from https://ftp-master.debian.org/keys.html and add them to `~/.gnupg/trustedkeys.gpg` (which is what **gpgv** uses by default).

```
gpg --no-default-keyring --keyring trustedkeys.gpg --import ziyi_key_2006.asc
```

- remover qualquer linha do `/etc/apt/sources.list` que não usa a estrutura normal de "dist", ou alterar o script para ele trabalhe com elas.

- estar preparado para ignorar o fato que o Debian security updates não assinou os Release files, e que os Sources files não tem os checksums apropriados no Release file (ainda).

- estar preparado para checar se as fontes estão assinadas com as chaves apropriadas.

This is the example code for **apt-check-sigs**, the latest version can be retrieved from http://people.debian.org/~ajt/apt-check-sigs. This code is currently in beta, for more information read http://lists.debian.org/debian-devel/2002/07/msg00421.html.

```
#!/bin/bash

# Copyright (c) 2001 Anthony Towns <ajt@debian.org>
#
# This program is free software; you can redistribute it and/or modify
# it under the terms of the GNU General Public License as published by
# the Free Software Foundation; either version 2 of the License, or
# (at your option) any later version.
#
# This program is distributed in the hope that it will be useful,
# but WITHOUT ANY WARRANTY; without even the implied warranty of
# MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
# GNU General Public License for more details.

rm -rf /tmp/apt-release-check
mkdir /tmp/apt-release-check || exit 1
```

```
cd /tmp/apt-release-check

 />OK
 />MISSING
 />NOCHECK
 />BAD

arch=`dpkg --print-installation-architecture`

am_root () {
        [ `id -u` -eq 0 ]
}

get_md5sumsize () {
        cat "$1" | awk '/^MD5Sum:/,/^SHA1:/' |
          MYARG="$2" perl -ne '@f = split /\s+/; if ($f[3] eq $ENV{"MYARG"}) {
print "$f[1] $f[2]\n"; exit(0); }'
}

checkit () {
        local FILE="$1"
        local LOOKUP="$2"

        Y="`get_md5sumsize Release "$LOOKUP"`"
        Y="`echo "$Y" | sed 's/^ *//;s/  */ /g'`"

        if [ ! -e "/var/lib/apt/lists/$FILE" ]; then
                if [ "$Y" = "" ]; then
                        # No file, but not needed anyway
                        echo "OK"
                        return
                fi
                echo "$FILE" >>MISSING
                echo "MISSING $Y"
                return
        fi
        if [ "$Y" = "" ]; then
                echo "$FILE" >>NOCHECK
                echo "NOCHECK"
                return
        fi
        X="`md5sum < /var/lib/apt/lists/$FILE | cut -d\  -f1` `wc -c < /var/lib
/apt/lists/$FILE`"
        X="`echo "$X" | sed 's/^ *//;s/  */ /g'`"
        if [ "$X" != "$Y" ]; then
                echo "$FILE" >>BAD
                echo "BAD"
                return
        fi
        echo "$FILE" >>OK
        echo "OK"
}

echo
```

```
echo "Checking sources in /etc/apt/sources.list:"
echo "~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~"
echo
(echo "You should take care to ensure that the distributions you're downloading
"
echo "are the ones you think you are downloading, and that they are as up to"
echo "date as you would expect (testing and unstable should be no more than"
echo "two or three days out of date, stable-updates no more than a few weeks"
echo "or a month)."
) | fmt
echo

cat /etc/apt/sources.list |
  sed 's/^ *//' | grep '^[^#]' |
  while read ty url dist comps; do
        if [ "${url%%:*}" = "http" -o "${url%%:*}" = "ftp" ]; then
                baseurl="${url#*://}"
        else
                continue
        fi

        echo "Source: ${ty} ${url} ${dist} ${comps}"

        rm -f Release Release.gpg
        lynx -reload -dump "${url}/dists/${dist}/Release" >/dev/null 2>&1
        wget -q -O Release "${url}/dists/${dist}/Release"

        if ! grep -q '^' Release; then
                echo "  * NO TOP-LEVEL Release FILE"
                >Release
        else
                origline=`sed -n 's/^Origin: *//p' Release | head -1`
                lablline=`sed -n 's/^Label: *//p' Release | head -1`
                suitline=`sed -n 's/^Suite: *//p' Release | head -1`
                codeline=`sed -n 's/^Codename: *//p' Release | head -1`
                dateline=`grep "^Date:" Release | head -1`
                dscrline=`grep "^Description:" Release | head -1`
                echo "  o Origin: $origline/$lablline"
                echo "  o Suite: $suitline/$codeline"
                echo "  o $dateline"
                echo "  o $dscrline"

                if [ "${dist%%/*}" != "$suitline" -a "${dist%%/*}" != "$codeline"
                        echo "  * WARNING: asked for $dist, got $suitline/$codelin
                fi

                lynx -reload -dump "${url}/dists/${dist}/Release.gpg" >/dev/null 2
                wget -q -O Release.gpg "${url}/dists/${dist}/Release.gpg"

                gpgv --status-fd 3 Release.gpg Release 3>&1 >/dev/null 2>&1 | sed
                        if [ "$gpgcode" = "GOODSIG" ]; then
                            if [ "$err" != "" ]; then
                                echo "  * Signed by ${err# } key: ${rest#* }"
                            else
```

```
                                    echo "  o Signed by: ${rest#* }"
                                    okay=1
                                fi
                                err=""
                            elif [ "$gpgcode" = "BADSIG" ]; then
                                echo "  * BAD SIGNATURE BY: ${rest#* }"
                                err=""
                            elif [ "$gpgcode" = "ERRSIG" ]; then
                                echo "  * COULDN'T CHECK SIGNATURE BY KEYID: ${rest %%
                                err=""
                            elif [ "$gpgcode" = "SIGREVOKED" ]; then
                                err="$err REVOKED"
                            elif [ "$gpgcode" = "SIGEXPIRED" ]; then
                                err="$err EXPIRED"
                            fi
                        done
                        if [ "$okay" != 1 ]; then
                            echo "  * NO VALID SIGNATURE"
                            >Release
                        fi)
                fi
                okaycomps=""
                for comp in $comps; do
                        if [ "$ty" = "deb" ]; then
                                X=$(checkit "`echo "${baseurl}/dists/${dist}/${comp}/binar
                                Y=$(checkit "`echo "${baseurl}/dists/${dist}/${comp}/binar
                                if [ "$X $Y" = "OK OK" ]; then
                                        okaycomps="$okaycomps $comp"
                                else
                                        echo "  * PROBLEMS WITH $comp ($X, $Y)"
                                fi
                        elif [ "$ty" = "deb-src" ]; then
                                X=$(checkit "`echo "${baseurl}/dists/${dist}/${comp}/sourc
                                Y=$(checkit "`echo "${baseurl}/dists/${dist}/${comp}/sourc
                                if [ "$X $Y" = "OK OK" ]; then
                                        okaycomps="$okaycomps $comp"
                                else
                                        echo "  * PROBLEMS WITH component $comp ($X, $Y)"
                                fi
                        fi
                done
                [ "$okaycomps" = "" ] || echo "  o Okay:$okaycomps"
                echo
        done

echo "Results"
echo "~~~~~~~"
echo

allokay=true

cd /tmp/apt-release-check
diff <(cat BAD MISSING NOCHECK OK | sort) <(cd /var/lib/apt/lists && find . -type
```

```
cd /tmp/apt-release-check
if grep -q ^ UNVALIDATED; then
    allokay=false
    (echo "The following files in /var/lib/apt/lists have not been validated."
    echo "This could turn out to be a harmless indication that this script"
    echo "is buggy or out of date, or it could let trojaned packages get onto"
    echo "your system."
    ) | fmt
    echo
    sed 's/^/    /' < UNVALIDATED
    echo
fi

if grep -q ^ BAD; then
    allokay=false
    (echo "The contents of the following files in /var/lib/apt/lists does not"
    echo "match what was expected. This may mean these sources are out of date,"
    echo "that the archive is having problems, or that someone is actively"
    echo "using your mirror to distribute trojans."
    if am_root; then
        echo "The files have been renamed to have the extension .FAILED and"
        echo "will be ignored by apt."
        cat BAD | while read a; do
            mv /var/lib/apt/lists/$a /var/lib/apt/lists/${a}.FAILED
        done
    fi) | fmt
    echo
    sed 's/^/    /' < BAD
    echo
fi

if grep -q ^ MISSING; then
    allokay=false
    (echo "The following files from /var/lib/apt/lists were missing. This"
    echo "may cause you to miss out on updates to some vulnerable packages."
    ) | fmt
    echo
    sed 's/^/    /' < MISSING
    echo
fi

if grep -q ^ NOCHECK; then
    allokay=false
    (echo "The contents of the following files in /var/lib/apt/lists could not"
    echo "be validated due to the lack of a signed Release file, or the lack"
    echo "of an appropriate entry in a signed Release file. This probably"
    echo "means that the maintainers of these sources are slack, but may mean"
    echo "these sources are being actively used to distribute trojans."
    if am_root; then
        echo "The files have been renamed to have the extension .FAILED and"
        echo "will be ignored by apt."
        cat NOCHECK | while read a; do
            mv /var/lib/apt/lists/$a /var/lib/apt/lists/${a}.FAILED
        done
```

```
    fi) | fmt
    echo
    sed 's/^/    /' < NOCHECK
    echo
fi

if $allokay; then
    echo 'Everything seems okay!'
    echo
fi

rm -rf /tmp/apt-release-check
```

Você pode precisar aplicar o seguinte patch para *sid* uma vez que **md5sum** adiciona um '-' após o sum quando a entrada é stdin:

```
@@ -37,7 +37,7 @@
        local LOOKUP="$2"

        Y="`get_md5sumsize Release "$LOOKUP"`"
-       Y="`echo "$Y" | sed 's/^ *//;s/  */ /g'`"
+       Y="`echo "$Y" | sed 's/-//;s/^ *//;s/  */ /g'`"

        if [ ! -e "/var/lib/apt/lists/$FILE" ]; then
                if [ "$Y" = "" ]; then
@@ -55,7 +55,7 @@
                return
        fi
        X="`md5sum < /var/lib/apt/lists/$FILE` `wc -c < /var/lib/apt/lists/$FILE`"
-       X="`echo "$X" | sed 's/^ *//;s/  */ /g'`"
+       X="`echo "$X" | sed 's/-//;s/^ *//;s/  */ /g'`"
        if [ "$X" != "$Y" ]; then
                echo "$FILE" >>BAD
                echo "BAD"
```

Release check of non Debian sources

Notice that, when using the latest apt version (with *secure apt*) no extra effort should be required on your part unless you use non-Debian sources, in which case an extra confirmation step will be required by apt-get. This is avoided by providing `Release` and `Release.gpg` files in the non-Debian sources. The `Release` file can be generated with **apt-ftparchive** (available in apt-utils 0.5.0 and later), the `Release.gpg` is just a detached signature. To generate both follow this simple procedure:

```
$ rm -f dists/unstable/Release
$ apt-ftparchive release dists/unstable > dists/unstable/Release
$ gpg --sign -ba -o dists/unstable/Release.gpg dists/unstable/Release
```

Esquema alternativo de assinatura per-package

The additional scheme of signing each and every packages allows packages to be checked when they are no longer referenced by an existing Packages file, and also third-party packages where no Packages ever existed for them can be also used in Debian but will not be default scheme.

This package signing scheme can be implemented using debsig-verify and debsigs. These two packages can sign and verify embedded signatures in the .deb itself. Debian already has the capability to do this now, but there is no feature plan to implement the policy or other tools since the archive signing scheme is prefered. These tools are available for users and archive administrators that would rather use this scheme instead.

Latest **dpkg** versions (since 1.9.21) incorporate a http://lists.debian.org/debian-dpkg/2001/03/msg00024.html that provides this functionality as soon as debsig-verify is installed.

NOTA: Atualmente `/etc/dpkg/dpkg.cfg` trabalha com "no-debsig" como padrão.

NOTA 2: Signatures from developers are currently stripped when they enter off the package archive since the currently preferred method is release checks as described previously.

Capítulo 8. Ferramentas de segurança no Debian

FIXME: Necessário mais conteúdo.

Debian fornece também uma série de ferramentas de segurança que podem tornar uma máquina com o sistema Debian adaptada para os propósitos de segurança. Estes propósitos incluem proteção dos sistemas de informação através de firewalls (de pacotes ou de aplicação), detecção de intrusão (baseados em rede e host), verificação de vulnerabilidades, antivirus, redes privadas, etc.

Desde o Debian 3.0 (*woody*), a distribuição caracteriza-se pelo software de criptografia integrado com a distribuição principal. OpenSSH e GNU Privacy Guard estão incluídos na instalação padrão, e criptografia forte está agora presente em navegadores e servidores Web, bancos de dados, e assim por diante. Além disso, a integração de criptografia está planejada para futuros lançamentos. Este software, devido as restrições de exportação nos EUA não foi distribuído com a distribuição principal, sendo disponível apenas em sites non-US.

Ferramentas de verificação remota de vulnerabilidades

As ferramentas fornecidas pelo Debian para realizar verificação remota de vulnerabilidade são: [1]

- nessus
- raccess
- nikto (**whisker**'s replacement)

A ferramenta mais completa e atualizada é, de longe, o nessus que é composta por um cliente (nessus) usado com uma GUI e um servidor (nessusd) que inicia os ataques programados. Nessus inclui verificação de vulnerabilidades remotas para a grande maioria de sistemas incluindo dispositivos de rede, servidores ftp e www, etc. Os últimos plugins de segurança tem a capacidade de analisar um sítio Web e tentar descobrir as páginas interativas disponíveis que podem ser atacadas. Existem também clientes Java e Win32 (não incluidas no Debian) que podem ser usados para acessar o servidor de gerenciamento.

Whisker é um varredor de verificação de vulnerabilidades Web, que inclui táticas anti-IDS (a maioria não são mais anti-IDS). É um dos melhores varreadores baseados em CGI disponíveis, sendo capaz de detectar servidores WWW e iniciar um dado conjunto de ataques contra ele. O banco de dados usado para a varredura pode ser facilmente modificado para fornecer novas informações.

Ferramentas de varredura de rede

Debian fornece algumas ferramentas usadas para a varredura remota de hosts (mas não para verificação de vulnerabilidades). Estas ferramentas são, em alguns casos, usadas pelos verificadores de vulnerabilidades como o primeiro tipo de "ataque" executado contra os hosts remotos na tentativa de determinar os serviços disponíveis. Atualmente Debian fornece os seguintes programas:

- nmap

[1] Algumas delas são fornecidas na instalação do pacote harden-remoteaudit.

- xprobe
- p0f
- knocker
- isic
- hping2
- icmpush
- nbtscan (for SMB /NetBIOS audits)
- fragrouter
- **strobe** (in the netdiag package)
- irpas

Enquanto o queso e o xprobe fornecem apenas detecção remota de sistema operacional (usando TCP/IP fingerprinting), nmap e knocker fazem, ambos, detecção de sistema operacional e varreadura de portas nos hosts remotos. Por outro lado, hping2 e icmpush podem ser usados nas técnicas de ataque ICMP remoto.

Desenvolvido especificamente para redes Netbios, nbtscan pode ser usado para varrer redes IP e recuperar informações de nome de servidores samba habilitados, incluindo nomes de usuários e de rede, endereços MAC...

Por outro lado, fragrouter pode ser usado para testar sistemas de detecção de instrusão e ver se o NIDS pode ser iludido com ataques de fragmentação.

FIXME: Verificar http://bugs.debian.org/153117 (ITP fragrouter) para ver se está incluído.

FIXME add information based on https://web.archive.org/web/20040725013857/http://www.giac.org/practical/gcux/Stephanie_Thomas_GCUX.pdf which describes how to use Debian and a laptop to scan for wireless (803.1) networks (link not there any more).

Auditoria Interna

Atualmente, somente a ferramenta tiger utilizada no Debian pode ser usada para executar auditorias internas de hosts (também chamadas de "caixa branca") de fato para determinar se o sistema de arquivos está corretamente configurado, que processos estão rodando no hosts, etc..

Auditoria de código fonte

Debian fornce três pacotes que podem ser utilizados para auditar códigos fontes em C/C++ e encontrar erros de programação que podem conduzir para potenciais falhas de segurança:

- flawfinder
- rats
- splint
- pscan

Redes Privadas Virtuais (VPN)

Uma rede privada virtual (VPN - Virtual Private Network) é um grupo de dois ou mais sistemas computacionais, tipicamente conectados a uma rede privada com acesso público de rede limitado, que se comunicam seguramente através de uma rede pública. VPNs podem conectar um simples computador a uma rede privada (cliente-servidor), ou uma LAN remota a uma rede privada (servidor-servidor). VPNs, muitas vezes, incluem o uso de criptografia, autenticação forte de usuários ou hosts remotos, e métodos para esconder a topologia da rede privada.

Debian fornece a maioria dos pacotes para configurar uma rede privada virtual criptografada:

- vtun
- tunnelv (non-US section)
- cipe-source, cipe-common
- tinc
- secvpn
- pptpd
- openvpn
- openswan (http://www.openswan.org/)

FIXME: Update the information here since it was written with FreeSWAN in mind. Check Bug #237764 and Message-Id: <200412101215.04040.rmayr@debian.org>.

O pacote OpenSWAN é provavelmente a melhor escolha, desde que ele promete interoperar com quase tudo que usa o protocolo IP seguro, IPSec (RFC 2411). Entretanto, os outros pacotes listados acima podem também ajudá-lo a ter um túnel seguro rapidamente. O protocolo de tunelamento ponto a ponto (PPTP) é um protocolo para VPN proprietário da Microsoft. É suportado no Linux, mas é conhecido por ter sérios problemas de segurança.

Para mais informações veja http://www.tldp.org/HOWTO/VPN-Masquerade-HOWTO.html (cobrindo IPSec e PPTP), http://www.tldp.org/HOWTO/VPN-HOWTO.html (cobrindo PPP sobre SSH), e http://www.tldp.org/HOWTO/mini/Cipe+Masq.html, e http://www.tldp.org/HOWTO/mini/ppp-ssh/index.html.

Também vale a pena verificar o http://yavipin.sourceforge.net/, mas este programa ainda não possue um pacote Debian disponível.

Tunelamento ponto a ponto

Se você deseja fornecer um servidor de tunelamento para um ambiente misto (com clientes Microsoft e Linux) e IPSec não é uma opção (desde que só é fornecido no Windows 2000 e Windows XP), você pode usar *PoPToP* (Servidor de Tunelamento Ponto a Ponto) disponível no pacote pptpd.

Se você deseja usar autenticação e criptografia da Microsoft com o servidor fornecido pelo pacote ppp, veja o seguinte trecho do FAQ:

```
O uso do PPP 2.3.8 só faz-se necessário se você deseja ter autenticação e
```

```
criptografia MSCHAPv2/MPPE compatíveis com a Microsoft. A razão para isto é que
o patch MSCHAPv2/MPPE atualmente aplicado (19990813) está sobre o PPP 2.3.8. Se
você não precisa de autenticação/criptografia compatível com a Microsoft,
qualquer versão 2.3.X do fonte do PPP será suficiente.
```

Entretanto, você também terá que aplicar o patch para o kernel fornecido no pacote kernel-patch-mppe, que contém o módulo pp_mppe para o pppd.

Saiba que a criptografia no ppptd força o armazenamento de senhas de usuários em texto limpo, e o protocolo MS-CHAPv2 contém http://mopo.informatik.uni-freiburg.de/pptp_mschapv2/.

Infra-estrutura de Chave Pública (PKI)

Infra-estrutura de Chave Pública (PKI - Public Key Infrastructure) é uma arquitetura de segurança introduzida para fornecer um nível adicional de confiança para trocas de informação em redes inseguras. Utiliza os conceitos de chaves de criptografia pública e privada para verificar a identidade de um remetente (assinatura) e para assegurar a privacidade (criptografia).

Quando considerar uma PKI, você encontrará uma variedade de situações:

- uma Autoridade Certificadora (CA - Certificate Authority) que pode distribuir e verificar certificados, e que pode trabalhar sobre uma dada hierarquia.
- um Diretório para manter certificados públicos de usuário
- um Banco de Dados (?) para manter Listas de Revogação de Certificados (CRL - Certificate Revocation Lists)
- dispositivos que interagem com a CA a fim de imprimir em smart cards/ tokens USB ou qualquer outra forma para armazenar seguramente os certificados.
- aplicações aptas a utilizarem certificados que podem usar certificados fornecidos por uma CA para realizar uma comunicação criptografada e verificar certificados dados contra CRL (para soluções de autenticação e assinatura de uma única vez completa)
- uma autoridade de marcação de tempo para assinar documentos digitalmente
- um console de gerenciamento a partir do qual tudo isso pode ser corretamente usado (geração de certificados, controle de lista de revogações, etc...)

Debian GNU/Linux tem pacotes de software para ajudar você com alguns desses pontos da PKI. Eles incluem **OpenSSL** (para geração de certificados), **OpenLDAP** (como um diretório para manter os certificados), **gnupg** e **openswan** (com suporte para o padrão X.509). Entretanto, como na versão Woody (Debian 3.0), Debian não tem nenhuma das autoridades certificadoras disponíveis gratuitamente como pyCA, http://www.openca.org ou os exemplos de CA do OpenSSL. Para mais informações, leia o http://ospkibook.sourceforge.net/.

Infra-estrutura SSL

Debian fornece alguns certificados SSL com a distribuição de modo que eles podem ser instalados localmente. Eles são encontrados no pacote ca-certificates, que fornece um repositório central dos certificados que foram submetidos para o Debian e aprovados (ou seja, verificados) pelo mantenedor do pacote e utéis para qualquer aplicação OpenSSL que verifica conexões SSL.

FIXME: leia o debian-devel para verificar se algo foi adicionado a ele.

Ferramentas Anti-vírus

Não existem muitas ferramentas anti-vírus incluídas no Debian GNU/Linux, provavelmente porque os usuários GNU/Linux não são aborrecidos com vírus. O modelo de segurança dos UN*X fazem uma distinção entre os processos privilegiados (root) e os processos de usuário, então quando um executável "hostil" é criado ou recebido por um usuário não-root e então executado, não pode "infectar" ou manipular o sistema em questão. Entretanto, worms e vírus no GNU/Linux existem, embora eles não tenham (ainda, esperançosamente) se espalhado em nenhuma distribuição Debian. Em qualquer caso, administradores podem querer construir gateways anti-vírus que os protejam contra vírus enviados para outros sistemas mais vulneráveis em suas redes.

Debian GNU/Linux atualmente fornece as seguintes ferramentas para a construção de ambientes anti-vírus:

- http://clamav.elektrapro.com/, fornecido no Debian *sarge* (futura versão 3.1). Pacotes são fornecidos tanto para o varredor de vírus (clamav), quanto para o daemon varredor (clamav-daemon) e para os arquivos de dados necessários para o varredor. Como a atualização do anti-vírus é crítica para o seu funcionamento, há duas formas diferentes de fazê-la: clamav-freshclam fornece um modo para atualização do banco de dados automaticamente através da Internet e clamav-data que fornece os arquivos de dados diretamente. [2]

- mailscanner um gateway de email com varredor de vírus e detector de spam. Usando o sendmail ou Exim como sua base, ele pode usar mais de 17 diferentes mecanismos de varredura de vírus (incluindo clamav)

- libfile-scan-perl que fornece File::Scan, uma extensão Perl para a varredura de arquivos em busca de vírus. Este módulo pode ser usado para fazer varredores de vírus independentes de plataforma.

- http://www.sourceforge.net/projects/amavis, fornecido no pacote amavis-ng e disponível no *sarge*, é um varredor de vírus em emails que é integrado com diferentes MTAs (Exim, Sendmail, Postfix, ou Qmail) e suporta cerca de quinze mecanismos de varredura de vírus (incluindo clamav, File::Scan e openantivirus).

- http://packages.debian.org/sanitizer, uma ferramenta que usa o pacote procmail que pode varrer anexos de email em busca de vírus, bloquear anexos baseados em seus nomes de arquivos e outras opções.

- http://packages.debian.org/amavis-postfix, um script que fornece uma interface de um agente de transporte de email para um ou mais varredores de vírus comerciais (este pacote é construído para suportar apenas o MTA **postfix**).

- exiscan, um varreador de e-mails escrito em Perl que funciona com o Exim.

- blackhole-qmail um filtro de spam para o Qmail que foi construído com suporte para o Clamav.

Alguns daemons de gateways já suportam extensões de ferramentas para construir ambientes anti-virus, incluindo exim4-daemon-heavy (a versão *pesada* do MTA Exim), frox (um servidor proxy e cache transparente para ftp), messagewall (um daemon proxy SMTP) e pop3vscan (um proxy transparente POP3).

Debian currently provide **clamav** as the only antivirus scanning software in the main official distribution and it also provides multiple interfaces to build gateways with antivirus capabilities for different protocols.

[2] Se você usar este último pacote e estiver usando um Debian oficial, o banco de dados não será atualizado com as atualizações de segurança. Você poderá usar o clamav-freshclam e o **clamav-getfiles** para gerar novos pacotes clamav-data ou atualizar do repositório do mantenedor, através da localização:

```
deb http://people.debian.org/~zugschlus/clamav-data/ / deb-src http://people.debian.org/~zugschlus/clamav-data/
```

Some other free software antivirus projects which might be included in future Debian GNU/Linux releases:http://sourceforge.net/projects/openantivirus/ (see http://bugs.debian.org/150698 and http://bugs.debian.org/150695).

FIXME: Is there a package that provides a script to download the latest virus signatures from http://www.openantivirus.org/latest.php?

FIXME: Verificar se scannerdaemon é o mesmo que o daemon varredor open anti-virus (ver ITPs).

However, Debian will *never* provide propietary (non-free and undistributable) antivirus software such as: Panda Antivirus, NAI Netshield, http://www.sophos.com/, http://www.antivirus.com, or http://www.ravantivirus.com. For more pointers see the http://www.computer-networking.de/~link/security/av-linux_e.txt. This does not mean that this software cannot be installed properly in a Debian system[3].

For more information on how to set up a virus detection system read Dave Jones' article https://web.archive.org/web/20120509212938/http://www.linuxjournal.com/article/4882.

Agentes GPG

É muito comum, atualmente, assinar digitalmente (e algumas vezes criptografar) e-mails. Você pode, por exemplo, verificar que muitas pessoas participando em listas de discussão assinam seus e-mails. Assinaturas de chave pública são atualmente o único mecanismo para verificar que um email foi enviado pelo remetente e não por qualquer outra pessoa.

Debian GNU/Linux fornece clientes de emails com funções embutidas para assinatura de emails que interagem com o gnupg ou pgp:

- evolution.
- mutt.
- kmail.
- icedove (rebranded version of Mozilla's Thunderbird) through the http://enigmail.mozdev.org/ plugin. This plugin is provided by the enigmail package.
- sylpheed. Dependendo de como a versão estável deste pacote evolua, você pode precisar usar a *versão bleeding edge*, sylpheed-claws.
- gnus, which when installed with the mailcrypt package, is an **emacs** interface to **gnupg**.
- kuvert, que fornece esta funcionalidade independentemente do agente de email do usuário (MUA - Mail User Agente) escolhido já que interage com o agente de transporte de email (MTA - Mail Transport Agente).

Servidores de chave permitem você fazer o download de chaves públicas publicadas que podem então verificar assinaturas. Um desses servidores de chaves é http://wwwkeys.pgp.net. gnupg pode automaticamente buscar chaves públicas que não estão em seu chaveiro público. Por exemplo, para configurar **gnupg** para usar o servidor de chaves acima, edite o arquivo `~/.gnupg/options` e adicione a seguinte linha: [4]

[3] Actually, there is an installer package for the *F-prot* antivirus, which is non-free but *gratis* for home users, called **f-prot-installer**. This installer, however, just downloads http://www.f-prot.com/products/home_use/linux/ and installs it in the system.
[4] Para mais exemplos de como configurar o **gnupg**, veja `/usr/share/doc/mutt/examples/gpg.rc`.

```
keyserver wwwkeys.pgp.net
```

A maioria dos servidores de chaves estão ligados, logo quando uma chave pública é adicionada em um servidor, esta é propagada para todos os outros servidores de chaves públicas. Existem também um pacote Debian debian-keyring, que fornece todas as chaves públicas dos desenvolvedores Debian. Os chaveiros do **gnupg** são instalados em `/usr/share/keyrings/`.

Para mais informações:

- http://www.gnupg.org/faq.html.
- http://www.gnupg.org/gph/en/manual.html.
- https://web.archive.org/web/20080201103530/http://www.dewinter.com/gnupg_howto/english/GPGMiniHowto.html.
- https://web.archive.org/web/20080513095235/http://www.uk.pgp.net/pgpnet/pgp-faq/.
- https://web.archive.org/web/20060222110131/http://www.cryptnet.net/fdp/crypto/gpg-party.html.

Capítulo 9. Developer's Best Practices for OS Security

This chapter introduces some best secure coding practices for developers writing Debian packages. If you are really interested in secure coding I recommend you read David Wheeler's http://www.dwheeler.com/secure-programs/ and http://www.securecoding.org by Mark G. Graff and Kenneth R. van Wyk (O'Reilly, 2003).

Best practices for security review and design

Developers that are packaging software should make a best effort to ensure that the installation of the software, or its use, does not introduce security risks to either the system it is installed on or its users.

In order to do so, they should make their best to review the source code of the package and detect any flaws that might introduce security bugs before releasing the software or distributing a new version. It is acknowledged that the cost of fixing bugs grows for different stages of its development, so it is easier (and cheaper) to fix bugs when designing than when the software has been deployed and is in maintenance mode (some studies say that the cost in this later phase is *sixty* times higher). Although there are some tools that try to automatically detect these flaws, developers should strive to learn about the different kind of security flaws in order to understand them and be able to spot them in the code they (or others) have written.

The programming bugs which lead to security bugs typically include: http://en.wikipedia.org/wiki/Buffer_overflow, format string overflows, heap overflows and integer overflows (in C/C++ programs), temporary http://en.wikipedia.org/wiki/Symlink_race (in scripts), http://en.wikipedia.org/wiki/Directory_traversal and command injection (in servers) and http://en.wikipedia.org/wiki/Cross_site_scripting, and http://en.wikipedia.org/wiki/SQL_injection (in the case of web-oriented applications). For a more complete information on security bugs review Fortify's http://vulncat.fortifysoftware.com/.

Some of these issues might not be easy to spot unless you are an expert in the programming language the software uses, but some security problems are easy to detect and fix. For example, finding temporary race conditions due to misuse of temporary directories can easily be done just by running `grep -r "/tmp/" .`. Those calls can be reviewed and replace the hardcoded filenames using temporary directories to calls to either **mktemp** or **tempfile** in shell scripts, File::Temp(3perl) in Perl scripts, or tmpfile(3) in C/C++.

There are a set of tools available to assist to the security code review phase. These include rats, flawfinder and pscan. For more information, read the http://www.debian.org/security/audit/tools.

When packaging software developers have to make sure that they follow common security principles, including:

- The software runs with the minimum privileges it needs:
  - The package does install binaries setuid or setgid. **Lintian** will warn of http://lintian.debian.org/reports/Tsetuid-binary.html, http://lintian.debian.org/reports/Tsetgid-binary.html and http://lintian.debian.org/reports/Tsetuid-gid-binary.html binaries.
  - The daemons the package provide run with a low privilege user (see “Creating users and groups for software daemons”)
- Programmed (i.e., **cron**) tasks running in the system do NOT run as root or, if they do, do not implement complex tasks.

If you have to do any of the above make sure the programs that might run with higher privileges have been audited for security bugs. If you are unsure, or need help, contact the http://www.debian.org/security/audit/. In the case of setuid/setgid binaries, follow the Debian policy section regarding http://www.debian.org/doc/debian-policy/ch-files.html#s10.9

For more information, specific to secure programming, make sure you read (or point your upstream to) http://www.dwheeler.com/secure-programs/ and the https://buildsecurityin.us-cert.gov/portal/ portal.

Creating users and groups for software daemons

If your software runs a daemon that does not need root privileges, you need to create a user for it. There are two kind of Debian users that can be used by packages: static uids (assigned by base-passwd, for a list of static users in Debian see “Usuários e grupos do sistema operacional”) and dynamic uids in the range assigned to system users.

In the first case, you need to ask for a user or group id to the base-passwd. Once the user is available there the package needs to be distributed including a proper versioned depends to the base-passwd package.

In the second case, you need to create the system user either in the *preinst* or in the *postinst* and make the package depend on `adduser (>= 3.11)`.

The following example code creates the user and group the daemon will run as when the package is installed or upgraded:

```
[...]
case "$1" in
  install|upgrade)

  # If the package has default file it could be sourced, so that
  # the local admin can overwrite the defaults

  [ -f "/etc/default/packagename" ] && . /etc/default/packagename

  # Sane defaults:

  [ -z "$SERVER_HOME" ] && SERVER_HOME=server_dir
  [ -z "$SERVER_USER" ] && SERVER_USER=server_user
  [ -z "$SERVER_NAME" ] && SERVER_NAME="Server description"
  [ -z "$SERVER_GROUP" ] && SERVER_GROUP=server_group

  # Groups that the user will be added to, if undefined, then none.
  ADDGROUP=""

  # create user to avoid running server as root
  # 1. create group if not existing
  if ! getent group | grep -q "^$SERVER_GROUP:" ; then
     echo -n "Adding group $SERVER_GROUP.."
     addgroup --quiet --system $SERVER_GROUP 2>/dev/null ||true
     echo "..done"
  fi
  # 2. create homedir if not existing
```

```
  test -d $SERVER_HOME || mkdir $SERVER_HOME
  # 3. create user if not existing
  if ! getent passwd | grep -q "^$SERVER_USER:"; then
    echo -n "Adding system user $SERVER_USER.."
    adduser --quiet \
            --system \
            --ingroup $SERVER_GROUP \
            --no-create-home \
            --disabled-password \
            $SERVER_USER 2>/dev/null || true
    echo "..done"
  fi
  # 4. adjust passwd entry
  usermod -c "$SERVER_NAME" \
          -d $SERVER_HOME   \
          -g $SERVER_GROUP  \
             $SERVER_USER
  # 5. adjust file and directory permissions
  if ! dpkg-statoverride --list $SERVER_HOME >/dev/null
  then
      chown -R $SERVER_USER:adm $SERVER_HOME
      chmod u=rwx,g=rxs,o= $SERVER_HOME
  fi
  # 6. Add the user to the ADDGROUP group
  if test -n $ADDGROUP
  then
      if ! groups $SERVER_USER | cut -d: -f2 | \
         grep -qw $ADDGROUP; then
           adduser $SERVER_USER $ADDGROUP
      fi
  fi
  ;;
  configure)

[...]
```

You have to make sure that the init.d script file:

- Starts the daemon dropping privileges: if the software does not do the setuid(2) or seteuid(2) call itself, you can use the `--chuid` call of **start-stop-daemon**.
- Stops the daemon only if the user id matches, you can use the **start-stop-daemon** `--user` option for this.
- Does not run if either the user or the group do not exist:

```
  if ! getent passwd | grep -q "^server_user:"; then
     echo "Server user does not exist. Aborting" >&2
     exit 1
  fi
  if ! getent group | grep -q "^server_group:" ; then
     echo "Server group does not exist. Aborting" >&2
     exit 1
  fi
```

If the package creates the system user it can remove it when it is purged in its *postrm*. This has some drawbacks, however. For example, files created by it will be orphaned and might be taken over by a new system user in the future if it is assigned the same uid[1]. Consequently, removing system users on purge is not yet mandatory and depends on the package needs. If unsure, this action could be handled by asking the administrator for the prefered action when the package is installed (i.e. through **debconf**).

Maintainers that want to remove users in their *postrm* scripts are referred to the **deluser**/**deluser** `--system` option.

Running programs with a user with limited privileges makes sure that any security issue will not be able to damage the full system. It also follows the principle of *least privilege*. Also consider you can limit privileges in programs through other mechanisms besides running as non-root[2]. For more information, read the http://www.dwheeler.com/secure-programs/Secure-Programs-HOWTO/minimize-privileges.html chapter of the *Secure Programming for Linux and Unix HOWTO* book.

[1] Some relevant threads discussing these drawbacks include http://lists.debian.org/debian-mentors/2004/10/msg00338.html and http://lists.debian.org/debian-devel/2004/05/msg01156.html
[2] You can even provide a SELinux policy for it

Capítulo 10. Antes do comprometimento do sistema

Keep your system secure

You should strive to keep your system secure by monitoring its usage and also the vulnerabilities that might affect it, patching them as soon as patches are available. Even though you might have installed a really secure system initially you have to remember that security in a system degrades with time, security vulnerabilities might be found for exposed system services and users might expose the system security either because of lack of understanding (e.g. accessing a system remotely with a clear-text protocol or using easy to guess passwords) or because they are actively trying to subvert the system's security (e.g. install additional services locally on their accounts).

Tracking security vulnerabilities

Although most administrators are aware of security vulnerabilities affecting their systems when they see a patch that is made available you can strive to keep ahead of attacks and introduce temporary countermeasures for security vulnerabilities by detecting when your system is vulnerable. This is specially true when running an exposed system (i.e. connected to the Internet) and providing a service. In such case the system's administrators should take care to monitor known information sources to be the first to know when a vulnerability is detected that might affect a critical service.

This typically includes subscribing to the announcement mailing lists, project websites or bug tracking systems provided by the software developers for a specific piece of code. For example, Apache users should regularly review Apache's http://httpd.apache.org/security_report.html and subscribe to the http://httpd.apache.org/lists.html#http-announce mailing list.

In order to track known vulnerabilities affecting the Debian distribution, the Debian Testing Security Team provides a https://security-tracker.debian.org/ that lists all the known vulnerabilities which have not been yet fixed in Debian packages. The information in that tracker is obtained through different public channels and includes known vulnerabilities which are available either through security vulnerability databases or http://www.debian.org/Bugs/. Administrators can search for the known security issues being tracked for https://security-tracker.debian.org/tracker/status/release/stable, https://security-tracker.debian.org/tracker/status/release/oldstable, https://security-tracker.debian.org/tracker/status/release/testing, or https://security-tracker.debian.org/tracker/status/release/unstable.

The tracker has searchable interfaces (by http://cve.mitre.org/ name and package name) and some tools (such as debsecan, see “Automatically checking for security issues with debsecan”) use that database to provide information of vulnerabilities affecting a given system which have not yet been addressed (i.e. those who are pending a fix).

Concious administrators can use that information to determine which security bugs might affect the system they are managing, determine the severity of the bug and apply (if available) temporary countermeasures before a patch is available fixing this issue.

Security issues tracked for releases supported by the Debian Security Team should eventually be handled through Debian Security Advisories (DSA) and will be available for all users (see “Atualizando continuamente o sistema”). Once security issues are fixed through an advisory they will not be available in the tracker, but you will be able to search security vulnerabilities (by CVE name) using the http://www.debian.org/security/crossreferences available for published DSAs.

Notice, however, that the information tracked by the Debian Testing Security Team only involves disclosed vulnerabilities (i.e. those already public). In some occasions the Debian Security Team might be handling and preparing DSAs for packages based on undisclosed information provided to them (for example, through closed vendor mailing lists or by upstream maintainers of software). So do not be surprised to find security issues that only show up as an advisory but never get to show up in the security tracker.

Atualizando continuamente o sistema

Você deve fazer as atualizações de segurança frequentemente. A grande maioria de exploits existentes é resultado de vulnerabilidades conhecidas que não foram corrigidas a tempo, como este http://www.cs.umd.edu/~waa/vulnerability.html (apresentando no IEEE Symposium on Security and Privacy em 2001) explica. Atualizações estão descritas em “Executar uma atualização de segurança”.

Verificando manualmente quais atualizações de segurança estão disponíveis

O Debian oferece uma ferramenta específica para verificar se o sistema precisa de atualização (veja o programa Tiger abaixo), mas muitos usuários preferem verificar manualmente se as atualizações de segurança estão disponíveis.

Se você configurou o seu sistema como descrito em “Executar uma atualização de segurança” você só precisa fazer:

```
# apt-get update
# apt-get upgrade -s
[ ... review packages to be upgraded ... ]
# apt-get upgrade
# checkrestart
[ ... restart services that need to be restarted ... ]
```

And restart those services whose libraries have been updated if any. Note: Read “Executar uma atualização de segurança” for more information on library (and kernel) upgrades.

O primeiro comando baixa a lista de pacotes disponíveis nos sources de pacotes configurados. A opção `-s` faz somente uma simulação, isto é, *não* baixa ou instala os pacotes e sim diz quais devem ser baixados/instalados. Você poderá saber que pacotes foram consertados pelo Debian e estão disponíveis para atualização. Por exemplo:

```
# apt-get upgrade -s
Reading Package Lists... Done
Building Dependency Tree... Done
2 packages upgraded, 0 newly installed, 0 to remove and 0  not upgraded.
Inst cvs (1.11.1p1debian-8.1 Debian-Security:3.0/stable)
Inst libcupsys2 (1.1.14-4.4 Debian-Security:3.0/stable)
Conf cvs (1.11.1p1debian-8.1 Debian-Security:3.0/stable)
Conf libcupsys2 (1.1.14-4.4 Debian-Security:3.0/stable)
```

In this example, you can see that the system needs to be updated with new cvs and cupsys packages which are being retrieved from *woody's* security update archive. If you want to understand why these packages are needed, you should go to http://security.debian.org and check which recent Debian Security Advisories have been published related to these packages. In this case, the related DSAs are https://lists.debian.org/debian-security-announce/2003/msg00014.html (for cvs) and https://lists.debian.org/debian-security-announce/2003/msg00013.html (for cupsys).

Notice that you will need to reboot your system if there has been a kernel upgrade.

Checking for updates at the Desktop

Since Debian 4.0 *lenny* Debian provides and installs in a default installation update-notifier. This is a GNOME application that will startup when you enter your Desktop and can be used to keep track of updates available for your system and install them. It uses update-manager for this.

In a stable system updates are only available when a security patch is available or at point releases. Consequently, if the system is properly configured to receive security updates as described in “Executar uma atualização de segurança” and you have a cron task running to update the package information you will be notified through an icon in the desktop notifcation area.

The notification is not intrusive and users are not forced to install updates. From the notification icon a desktop user (with the administrator's password) can access a simple GUI to show available updates and install them.

This application works by checking the package database and comparing the system with its contents. If the package database is updated periodically through a **cron** task then the contents of the database will be newer than the packages installed in the system and the application will notify you.

Apt installs such a task (`/etc/cron.d/apt`) which will run based on Apt's configuration (more specifically *APT::Periodic*). In the GNOME environment this configuration value can be adjusted by going to System > Admin > Software origins > Updates, or running **/usr/bin/software-properties**.

If the system is set to download the packages list daily but not download the packages themselves your `/etc/apt/apt.conf.d/10periodic` should look like this:

```
APT::Periodic::Update-Package-Lists "1";
APT::Periodic::Download-Upgradeable-Packages "0";
```

You can use a different cron task, such as the one installed by cron-apt (see “Verificando automaticamente por atualizações com o cron-apt”). You can also just manually check for upgrades using this application.

Users of the KDE desktop environment will probably prefer to install adept and adept-notifier instead which offers a similar functionality but is not part of the standard installation.

Verificando automaticamente por atualizações com o cron-apt

Another method for automatic security updates is the use of cron-apt. This package provides a tool to update the system at regular intervals (using a cron job), and can also be configured to send mails to the system administrator using the local mail transport agent. It will just update the package list and download new packages by default but it can be configured to automatically install new updates.

Note que você pode querer verificar a versão da distribuição, como descrito em “Per distribution release check”, se você pretende atualizar automaticamente o seu sistema (mesmo somente baixando pacotes). Caso contrário você não terá certeza que os pacotes baixados realmente são de origem confiável.

More information is available at the http://www.debian-administration.org/articles/162.

Automatically checking for security issues with debsecan

The **debsecan** program evaluates the security status of by reporting both missing security updates and security vulnerabilities. Unlike cron-apt, which only provides information related to security updates available, but this tool obtains information from the security vulnerability database maintained by the Debian Security Team which includes also information on vulnerabilities which are not yet fixed through

a security update. Consequently, it is more efficient at helping administrators track security vulnerabilities (as described in “Tracking security vulnerabilities”).

Upon installing the Debian package debsecan, and if the administrator consents to it, it will generate a cron task that will make it run and send the output to a specific user whenever it finds a vulnerable package. It will also download the information from the Internet. The location of the security database is also part of the questions ask on installation and are later defined `/etc/default/debsecan`, it can be easily adjusted for systems that do not have Internet access so that they all pull from a local mirror so that there is a single point that access the vulnerability database.

Notice, however, that the Security Team tracks many vulnerabilities including low-risk issues which might not be fixed through a security update and some vulnerabilities initially reported as affecting Debian might, later on, upon investigation, be dismissed. **Debsecan** will report on all the vulnerabilities, which makes it a quite more verbose than the other tools described above.

More information is available at the http://www.enyo.de/fw/software/debsecan/.

Outros métodos para atualizações de segurança

There is also the apticron, which, similarly to cron-apt will check for updates and send mails to the administrator. More information on apticron is available at the http://www.debian-administration.org/articles/491.

You might also want to take a look at http://clemens.endorphin.org/secpack/ which is an unofficial program to do security updates from security.debian.org with signature checking written by Fruhwirth Clemens. Or to the Nagios Plugin http://www.unixdaemon.net/nagios_plugins.html#check_debian_packages written by Dean Wilson.

Evite usar versões instáveis

Ao menos que você tenha tempo para aplicar patches de segurança toda vez que uma vulnerabilidade é descoberta, você *não* deve usar a versão instável do Debian para sistemas em produção. A principal razão para isto é que não há atualizações de segurança para a versão *unstable*.

O fato é que algumas questões relacionadas à segurança podem surgir na distribuição instável e *não* na *stable*. Isto porque novas funcionalidades são constantemente adicionadas às aplicações, assim como novas aplicações são incluídas sem serem totalmente testadas.

Para se fazer atualizações de segurança na versão *unstable*, você pode fazer uma atualização completa para nova versão (que atualiza muito mais do que somente os pacotes afetados). Embora existam algumas exceções, patches de segurança geralmente só são portadas para a versão *stable*. A idéia principal é que entre as atualizações, *nenhum código novo* deve ser adicionado, somente consertos para questões importantes.

Notice, however, that you can use the security tracker (as described in “Tracking security vulnerabilities”) to track known security vulnerabilities affecting this branch.

Security support for the testing branch

Se você estiver utilizando uma versão em *testing*, existem algumas questões relacionadas à disponibilidade das atualizações de segurança que devem ser levadas em conta:

- Quando um conserto de segurança é preparado, o Time de Segurança lança o patch para a versão *stable* (desde que a estável é geralmente algumas versões menor ou maior atrás). Os mantenedores de pacotes são responsáveis por preparar o patch para a versão *unstable*, geralmente baseado nos novos lançamentos. Algumas vezes as alterações acontecem quase ao mesmo tempo e em outras um dos

lançamentos disponibiliza o conserto de segurança antes. Os pacotes para a distribuição *stable* são testados bem mais a fundo do que para a *unstable*, já que esta irá fornecer na maioria dos casos a última versão do lançamento (que pode incluir novos e desconhecidos bugs)

- Atualizações de segurança estão disponíveis para a versão *unstable* geralmente quando os mantenedores fazem um novo pacote e para a versão *stable* quando o Time de Segurança publica um DSA e faz um novo upload. Observe que nada disso altera a versão em *testing*.

- Se nenhum (novo) bug é detectado na versão *unstable* do pacote, ele passa para a versão em *testing* depois de algum tempo. Este tempo geralmente é de dez dias, embora dependa de algumas coisas como a prioridade de upload e se o pacote está ou não bloqueado para entrar em teste por causa de dependências. Note que se o pacote estiver bloqueado, a prioridade de upload não afetará o tempo que ele leva para entrar na versão em teste.

Esse comportamento pode ser alterado conforme o estado de lançamento da distribuição. Quando uma distribuição está perto de ser lançada, o Time de Segurança ou os mantenedores dos pacotes devem fornecer atualizações de segurança diretamente para a versão em teste.

Additionally, the http://secure-testing-master.debian.net can issue Debian Testing Security Advisories (DTSAs) for packages in the *testing* branch if there is an immediate need to fix a security issue in that branch and cannot wait for the normal procedure (or the normal procedure is being blocked by some other packages).

Users willing to take advantage of this support should add the following lines to their `/etc/apt/sources.list` (instead of the lines described in “Executar uma atualização de segurança”):

```
    deb http://security.debian.org testing/updates main contrib non-free
# This line makes it possible to donwload source packages too
    deb-src  http://security.debian.org testing/updates main contrib non-free
```

For additional information on this support please read the http://lists.debian.org/debian-devel-announce/2006/05/msg00006.html. This support officially started in http://lists.debian.org/debian-devel-announce/2005/09/msg00006.html in a separate repository and was later integrated into the main security archive.

Atualizações automáticas no sistema Debian GNU/Linux

Primeiro de tudo, atualizações automáticas não são recomendadas, já que o administrador deve revisar os DSAs (alertas de segurança do Debian) e entender o impacto causado pela atualização de segurança no sistema.

Para atualizar o seu sistema automaticamente você deve:

- Configure **apt** so that those packages that you do not want to update stay at their current version, either with **apt**'s *pinning* feature or marking them as *hold* with **aptitude** or **dpkg**.

  To pin the packages under a given release, you must edit `/etc/apt/preferences` (see apt_preferences(5)) and add:

  ```
  Package: *
  Pin: release a=stable
  Pin-Priority: 100
  ```

  FIXME: verifcar se a configuração está OK.

- Either use cron-apt as described in “Verificando automaticamente por atualizações com o cron-apt” and enable it to install downloaded packages or add a **cron** entry yourself so that the update is run daily, for example:

```
apt-get update && apt-get -y upgrade
```

  A opção `-y` faz com que o **apt** assuma 'sim' para todos os prompts que aparecerão durante a atualização. Em alguns casos, é melhor você usar a opção `--trivial-only` em vez de `--assume-yes` (equivalente a `-y`). [1]

- Configure o **cron** para que o **debconf** não faça nenhuma pergunta durante as atualizações, funcionando de forma não-interativa. [2]

- Verifique os resultados da execução do **cron**, que enviará um mail para o superusuário (ao menos que a variável de ambiente `MAILTO` seja alterada no script).

Uma alternativa mais segura seria usar a opção `-d` (ou `--download-only`), que irá fazer o download dos pacotes necessários mas não os instalará. Então se a execução do **cron** mostrar que o sistema precisa ser atualizado, esta atualização pode ser feita manualmente.

E para finalizar estas tarefas, o sistema deve ser configurado apropriadamente para fazer o download das atualizações de segurança como discutido no “Executar uma atualização de segurança”.

Entretanto, isto não é recomendado para a versão *unstable* sem que haja uma análise cuidadosa, uma vez que pode tornar o seu sistema inutilizável se algum pacote importante que estiver com um bug sério for instalado. A *testing* é um pouco mais *segura* com relação a isto, já que os bugs sérios podem ser detectados antes do pacote ser movido para a versão em teste (embora, você *não* tenha atualizações de segurança disponíveis para todos).

Se você tem uma distribuição mista, isto é, uma instalação *stable* com alguns pacotes atualizados para a versão em *testing* ou *unstable*, você pode utilizar o recurso de *pinning* assim como a opção `--target-release` do **apt** para atualizar *somente* aqueles pacotes que devem ser atualizados. [3]

Faça verificações de integridade periódicas

A verificação de integridade é feita baseada na informação completa do sistema gerada depois da instalação (ex. o *snapshot* descrito em “Fazendo um snapshot do sistema”) e deve ser feita de tempos em tempos. Com a verificação de integridade é possível detectar modificações no sistema de arquivos feitas por um intruso ou por algum erro do administrador do sistema.

As verificações de integridade devem ser, se possível, feitas offline [4] . Isto é, utilizar outro sistema operacional para fazer a verificação, evitando assim um falso senso de segurança (ex. falsos negativos) produzido por, por exemplo, rootkits instalados. A base de dados de integridade verificada pelo sistema também deve ser usada em uma mídia somente leitura.

Você deve considerar fazer a verificação online utilizando qualquer ferramenta de verificação de integridade do sistema de arquivos disponíveis (descrito em “Verificando a integridade do sistema de

[1] Você também pode optar por usar a opção `--quiet` (`-q`) para diminuir a quantidade de informações de saída do **apt-get**. Caso nenhum pacote esteja sendo instalado, nenhuma informação é mostrada na tela.
[2] Note que alguns pacotes podem *não* usar o **debconf** e a atualizações irão parar para que o usuário entre com alguma configuração.
[3] Isso é uma prática comum, já que muitos usuários preferem manter o sistema estável, podendo atualizar alguns pacotes para a versão *unstable* para obter novas funcionalidades. Esta necessidade surge devido ao desenvolvimento de alguns projetos ser mais rápido que o tempo gasto entre os lançamentos da versão *stable* do Debian.
[4] Uma maneira fácil de fazer isso é utilizar um Live CD, tipo o http://www.knoppix-std.org/ que inclue ambas as ferramentas de verificação de arquivos e a base de dados de integridade do seu sistema.

arquivos"), se você não puder deixar o sistema fora do ar. Entretanto, algumas precauções devem ser levadas em conta como a utilização de uma base de dados da integridade somente para leitura e assegurar que a ferramenta de verificação de integridade (e o kernel do sistema operacional) não esteja sendo usada.

Algumas das ferramentas citadas nesta seção, como **aide**, **integrit** ou **samhain** já estão preparadas para fazer revisões periódicas (através do crontab nas duas primeiras e através de um daemon standalone na **samhain**) e pode avisar o administrador por diferentes canais (geralmente e-mail, mas **samhain** também pode enviar pages, traps SNMP ou alertas do syslog) quando ocorrem alterações no sistema de arquivos.

Claro que se você for executar uma atualização do sistema, deve ser tirado novamente um snapshot para acomodar as alterações sofridas durante a atualização de segurança.

Configure um sistema de Detecção de Intrusão

O Debian GNU/Linux inclue ferramentas para detecção de intrusão, que é nada mais do que a prática de detectar atividades impróprias ou maliciosas no seu sistema local, ou outros sistemas que estejam na sua rede privada. Este tipo de defesa é importante se o sistema for altamente crítico ou você for realmente paranóico. Os tipos mais comuns de detecção de intrusão são detecção estatística de anomalias e detecção baseada em algum padrão.

Sempre tenha em mente que para melhorar a segurança do sistema com a instalação de uma dessas ferramentas, você deve ter um mecanismo de alertas e respostas elaborado. Detecção de intrusão é perda de tempo se você não for alertar ninguém.

Quando um ataque em particular for detectado, a maioria das ferramentas de detecção de intrusão irá tanto gerar um log do evento com o **syslogd** enviar um e-mail para o super-usuário (o destinatário geralmente é configurável). Um administrador precisa configurar propriamente as ferramentas para que falsos positivos não gerem alertas. Alertas também devem informar um ataque que pode estar acontecendo e ele não será útil, digamos, um dia depois que ocorrer. Então tenha certeza que existe uma política apropriada para tratar os alertas e que os mecanimos técnicos para implementar essa política sejam viáveis.

An interesting source of information is http://www.cert.org/tech_tips/intruder_detection_checklist.html

Detecção de intrusão baseada em rede

As ferramentas de detecção de intrusão baseada em rede monitoram o tráfego em um segmento de rede e utilizam essas informações como fonte dos dados para serem analisados. Especificamente, os pacotes da rede são examinados, e eles são verificados para ver se existe uma certa assinatura de pacotes maliciosos.

snort is a flexible packet sniffer or logger that detects attacks using an attack signature dictionary. It detects a variety of attacks and probes, such as buffer overflows, stealth port scans, CGI attacks, SMB probes, and much more. **snort** also has real-time alerting capability. You can use **snort** for a range of hosts on your network as well as for your own host. This is a tool which should be installed on every router to keep an eye on your network. Just install it with `apt-get install snort`, follow the questions, and watch it log. For a little broader security framework, see http://www.prelude-ids.org.

Debian's snort package has many security checks enabled by default. However, you should customize the setup to take into account the particular services you run on your system. You may also want to seek additional checks specific to these services.

There are other, simpler tools that can be used to detect network attacks. portsentry is an interesting package that can tip you off to port scans against your hosts. Other tools like ippl or iplogger will also detect some IP (TCP and ICMP) attacks, even if they do not provide the kind of advanced techniques **snort** does.

You can test any of these tools with the Debian package idswakeup, a shell script which generates false alarms, and includes many common attack signatures.

Detecção de intrusão baseada em host

A detecção de intrusão baseada em host envolve o carregamento de um software no sistema a ser monitorado e que utiliza arquivos de log e/ou os programas de auditoria de sistema como uma fonte de dados. Ele procura por processos suspeitos, monitora acesso ao host e pode até monitorar alterações em arquivos críticos do sistema.

tiger is an older intrusion detection tool which has been ported to Debian since the Woody branch. **tiger** provides checks of common issues related to security break-ins, like password strength, file system problems, communicating processes, and other ways root might be compromised. This package includes new Debian-specific security checks including: MD5sums checks of installed files, locations of files not belonging to packages, and analysis of local listening processes. The default installation sets up **tiger** to run each day, generating a report that is sent to the superuser about possible compromises of the system.

Log analysis tools, such as logcheck can also be used to detect intrusion attempts. See “Using and customizing **logcheck**”.

Em adição, pacotes que monitoram a integridade do sistema de arquivo (veja “Verificando a integridade do sistema de arquivos”) podem ser perfeitamente úteis na detecção de anomalias em um ambiente seguro. É muito provável que uma intrusão efetiva irá modificar alguns arquivos no sistema de arquivo local para driblar a política de segurança local, instalar Trojans, ou criar usuários. Tais eventos podem ser detectados com os programas para verificação de integridade do arquivo.

Evitando os rootkits

Loadable Kernel Modules (LKM)

Loadable kernel modules são arquivos contendo componentes carregados dinamicamente no kernel e são usados para expandir a funcionalidade do mesmo. O benefício principal de se usar módulos é a habilidade de adicionar dispositivos adicionas, como uma placa de rede Ethernet ou uma placa de som, sem ter que aplicar um patch no código-fonte e recompilar todo o kernel. Entretanto, os crackers vêm usando os LKMs para criar rootkits (knark e adore), abrindo backdoors nos sistemas GNU/Linux.

Os backdoors LKM estão cada vez mais sofisticados e mais difíceis de serem detectados que os rootkits tradicionais. Eles podem esconder processos, arquivos, diretórios e até mesmo conexões sem precisar modificar o código fonte dos binários. Por exemplo, um LKM malicioso pode forçar o kernel a esconder processos específicos do `procfs`, então mesmo uma cópia original do binário **ps** pode não listar informações precisas sobre os processos que estão rodando no sistema.

Detectando rootkits

Existem duas estratégias para defender seu sistema de rootkits LKM, a defesa pró-ativa e a reativa. O trabalho de detecção pode ser simples e fácil, ou difícil e cansativo, dependendo da estratégia escolhida.

Defesa pró-ativa

A vantagem para este tipo defesa é que ela previne qualquer dano ao sistema logo de início. Uma estratégia para esse tipo de defesa é conhecida como *pegar eles primeiro*, que é carregar na memória um módulo LKM designado para proteger o sistema de outros LKMs maliciosos. A segunda estratégia é remover algumas funcionalidades do próprio kernel. Por exemplo, você pode desabilitar a opção de carrergar módulos no kernel. Entretanto, note que existem rootkits que podem funcionar até mesmo neste caso. Alguns deles podem mexer com o `/dev/kmem` (memória do kernel) diretamente para torná-los indetectáveis.

O Debian GNU/Linux tem poucos pacotes que podem ser usados para montar uma defesa pró-ativa:

lcap - A user friendly interface to remove *capabilities* (kernel-based access control) in the kernel, making the system more secure. For example, executing `lcap CAP_SYS_MODULE`[5] will remove module loading capabilities (even for the root user).[6] There is some (old) information on capabilities at Jon Corbet's http://lwn.net/1999/1202/kernel.php3 section on LWN (dated December 1999).

Se você realmente não precisa de muitos recursos do kernel no seu sistema GNU/Linux, você pode desabilitar o suporte aos módulos carregáveis durante a configuração do kernel. Para desabilitar este suporte, somente altere o CONFIG_MODULES=n durante o estágio de configuração da construção do seu kernel, ou no arquivo `.config`. Isto irá prevenir os rootkits LKM, mas você irá perder esta funcionalidade poderosa no kernel do Linux. Desabilitar a opção para carregar módulos no kernel pode muitas vezes sobrecarregar o kernel. Neste caso, é melhor deixar o kernel com o suporte.

Defesa reativa

A vantagem da defesa reativa é que ela não consome os recursos do sistema. Ela trabalha comparando a tabela de chamadas ao sistema com uma cópia autêntica conhecida, o arquivo em disco `System.map`. Claro que a defesa reativa somente notificará ao administrador do sistema depois que o sistema já estiver sido comprometido.

Detection of some root-kits in Debian can be accomplished with the chkrootkit package. The http://www.chkrootkit.org program checks for signs of several known root-kits on the target system, but is not a definitive test.

Idéias Geniais/Paranóicas — o que você pode fazer

Esta é provavelmente a mais instável e divertida seção, apenas espero que algumas das ideias "duh, isso parece loucura" possam ser realizadas. A seguir algumas idéias para melhorar a segurança — talvez geniais, paranóicas, loucas ou até inspiradas dependendo do seu ponto de vista.

- Brincando com o PAM. Como citado no artigo Phrack 56 PAM, a coisa legal do PAM é que "Você é limitado somente pelo o que pode imaginar". É verdade. Imagine efetuar login de root somente através de impressão digital ou verificação de retina ou cartão de criptografia (por que usei a conjunção OU em vez de E?).

- Gravação fascista de logs. Eu prefiro me referir à toda discussão anterior acima como um "esquema leve de logs". Se você quiser fazer um esquema real de logs, pegue uma impressora com papel de formulário contínuo, e envie todos os logs para ela. Parece engraçado, mas é realmente confiável e as informações não podem ser sobrescritas ou apagadas.

- Distribuição de CD. Essa idéia é muito simples de se realizar e oferece uma boa segurança. Crie uma distribuição Debian segura, com as regras de firewall apropriadas. Coloque ela em uma imagem ISO inicializável e grave em um CDROM. Agora você tem uma distribuição somente leitura, com mais ou menos 600 MB de espaço para os serviços. Tenha certeza de que todos os dados que devem ser escritos

[5] There are over 28 capabilities including: `CAP_BSET`, `CAP_CHOWN`, `CAP_FOWNER`, `CAP_FSETID`, `CAP_FS_MASK`, `CAP_FULL_SET`, `CAP_INIT_EFF_SET`, `CAP_INIT_INH_SET`, `CAP_IPC_LOCK`, `CAP_IPC_OWNER`, `CAP_KILL`, `CAP_LEASE`, `CAP_LINUX_IMMUTABLE`, `CAP_MKNOD`, `CAP_NET_ADMIN`, `CAP_NET_BIND_SERVICE`, `CAP_NET_RAW`, `CAP_SETGID`, `CAP_SETPCAP`, `CAP_SETUID`, `CAP_SYS_ADMIN`, `CAP_SYS_BOOT`, `CAP_SYS_CHROOT`, `CAP_SYS_MODULE`, `CAP_SYS_NICE`, `CAP_SYS_PACCT`, `CAP_SYS_PTRACE`, `CAP_SYS_RAWIO`, `CAP_SYS_RESOURCE`, `CAP_SYS_TIME`, and `CAP_SYS_TTY_CONFIG`. All of them can be de-activated to harden your kernel.

[6] You don't need to install lcap to do this, but it's easier than setting `/proc/sys/kernel/cap-bound` by hand.

sejam feitos pela rede. É impossível para um intruso ter acesso de leitura/escrita no sistema, e qualquer alteração feita pelo intruso pode ser desfeita em uma reinicilização do sistema.

- Desabilite a capacidade de carregar módulos. Como discutido anteriormente, quando você desabilita o uso de módulos em tempo de compilação do kernel, muitos backdoors baseados em kernel ficam impossíveis de serem implementados, pois a maioria deles é baseada na instalação de módulos do kernel modificados.

- Grave os logs por um cabo serial. (contribuido por Gaby Schilders) Já que os servidores ainda têm portas serial, imagine ter um sistema de gravação de logs para um série de servidores. O sistema de logs é desconectado da rede, e conectado aos servidores via um multiplexador de porta serial (Cyclades ou algo do tipo). Agora faça com que todos os seus servidores gravem o log através da porta serial. A máquina de log vai somente aceitar o texto plano como entrada nas portas serial e escrever em um arquivo de log. Conecte um gravador de CD/DVD e grave o arquivo de log quando atingir a capacidade máxima da mídia.

- Altere as atribuições do arquivo usando **chattr**. (dica tirada do Tips-HOWTO, escrito por Jim Dennis). Depois de uma instalação limpa e configuração inicial, use o programa **chattr** com o atributo `+i` para que os arquivos não sejam modificados (o arquivo não pode ser apagado, renomeado, criado link ou escrito algo nele). Defina este atributo em todos os arquivos que estão em `/bin`, `/sbin/`, `/usr/bin`, `/usr/sbin`, `/usr/lib` e também nos arquivos do kernel no root. Você também pode fazer uma cópia de todos os arquivos do `/etc/`, usando o **tar** ou algo do tipo e marcar o arquivo comprimido como imutável.

  Esta estratégia irá ajudar a limitar o estrago que você poderá causar estando logado como root. Você não poderá sobrescrever arquivos por engano, nem deixar o sistema inoperante digitando por engano um espaço no comando **rm -fr** (você pode ainda fazer um monte de estragos no seus dados — mas suas bibliotecas e seus binários estarão seguros.)

  Esta estratégia também faz com que uma variedade de exploits de segurança e de negação de serviços (DoS) sejam difíceis ou impossíveis de serem realizados (já que a maioria deles conta com a permissão de sobrescrever um arquivo através de algum programa SETUID que a princípio *não esteja fornecendo um comando shell arbitrário*).

  Uma inconveniência desse tipo de estratégia aparece durante a compilação e instalação de alguns binários do sistema. Por outro lado, isso previne que um comando **make install** sobrescreva os arquivos. Quando você se esquece de ler o Makefile e executa um **chattr -i** nos arquivos a serem sobrescritos, (também nos diretórios nos quais serão adicionados os arquivos) # o comando make falha. Então você deve usar o comando **chattr** para desativar a flag de imutável e rodar o make novamente. Você também pode optar por mover os binários e as bibliotecas antigas para dentro de um diretório .old/ ou para um arquivo tar por exemplo.

  Note que esta estratégia também impede que você atualize seu próprio sistema de pacotes, já que os arquivos que os pacotes a serem atualizados fornecem não podem ser sobrescritos. Você pode fazer um script ou usar outro mecanismo parecido para desativar a permissão de imutável em todos os binários antes de fazer um **apt-get update**.

- Você pode brincar um pouco com o cabeamento UTP cortando 2 ou 4 fios, tornando um cabo de tráfego unidirecional. Então use pacotes UDP para enviar informação para uma máquina de destino que atuaria como um servidor de log seguro ou até mesmo um sistema de armazenamento de cartões de crédito.

Construindo um honeypot

Um honeypot é um sistema feito para auxiliar os administradores de sistemas a descobrir como os crackers sondam a máquina em busca de exploits. O sistema é configurado com a expectativa e objetivo de ser

sondado, atacado e potencialmente invadido. Aprendendo as ferramentas e os métodos empregados pelo cracker, um administrador de sistema pode saber como melhor proteger seus sistemas e a rede.

Um sistema Debian GNU/Linux pode ser facilmente configurado como um honeypot, se você dedicar tempo para implementar e monitorá-lo. Simplesmente configure o servidor falso com um firewall e algumas ferramentas de detecção de intrusão de rede, coloque ele na Internet, e espere. Tome o cuidado de que se o sistema for invadido você seja imediatamente alertado (veja “A importância dos logs e alertas”), desta forma você poderá tomar providências necessárias e paralizar a invasão quando tiver informações suficientes. Abaixo estão alguns dos pacotes e questões importantes quando estiver configurando seu honeypot:

- A tecnologia de firewall que irá usar (fornecidade pelo kernel do Linux).
- syslog-ng, useful for sending logs from the honeypot to a remote syslog server.
- snort, to set up capture of all the incoming network traffic to the honeypot and detect the attacks.
- osh, a SETUID root, security enhanced, restricted shell with logging (see Lance Spitzner's article below).
- Of course, all the daemons you will be using for your fake server honeypot. Depending on what type of attacker you want to analyse you will or will *not* harden the honeypot and keep it up to date with security patches.
- Integrity checkers (see “Verificando a integridade do sistema de arquivos”) and The Coroner's Toolkit (tct) to do post-attack audits.
- honeyd and farpd to setup a honeypot that will listen to connections to unused IP addresses and forward them to scripts simulating live services. Also check out iisemulator.
- tinyhoneypot to setup a simple honeypot server with fake services.

If you cannot use spare systems to build up the honeypots and the network systems to protect and control it you can use the virtualisation technology available in **xen** or **uml** (User-Mode-Linux). If you take this route you will need to patch your kernel with either kernel-patch-xen or kernel-patch-uml.

You can read more about building honeypots in Lanze Spitzner's excellent article http://www.net-security.org/text/articles/spitzner/honeypot.shtml (from the Know your Enemy series). Also, the http://project.honeynet.org/ provides valuable information about building honeypots and auditing the attacks made on them.

Capítulo 11. Depois do comprometimento do sistema (resposta a incidentes)

Comportamento comum

Se você estiver fisicamente presente quando o ataque ocorrer, sua primeira obrigação é tirar a máquina da rede desconectando o cabo de rede da placa (se isso não for influenciar nas transações dos negócios). Desativando a rede na camada 1 é a única forma de manter o invasor longe da máquina comprometida (conselho sábio de Philip Hofmesiter).

Entretanto, alguns rootkits ou back doors são capazes de detectar este tipo de evento e reagir a ele. Ver um `rm -rf /` sendo executado quando você desativa a rede não é muito engraçado. Se você se nega a correr o risco e tem certeza que o sistema foi comprometido, você deve *desconectar o cabo de energia* (todos eles se existirem mais de um) e cruzar os dedos. Isso pode ser extremo mas, de fato, irá evitar qualquer bomba lógica que o invasor possa ter programado. Nesses casos, o sistema comprometido *não deve ser reiniciado*. Os discos rígidos também devem ser colocados em outro sistema para serem analisados, ou deve ser usado outro tipo de mídia (um CD-ROM) para inicializar o sistema e analisá-lo. Você *não* deve usar os discos de recuperação do Debian para inicializar o sistema, mas você *pode* utilizar o shell fornecido pelos discos de instalação (use Alt+F2 para acessá-lo) para analisar o sistema. [1]

The most recommended method for recovering a compromised system is to use a live-filesystem on CD-ROM with all the tools (and kernel modules) you might need to access the compromised system. You can use the mkinitrd-cd package to build such a CD-ROM[2]. You might find the http://www.caine-live.net/ (Computer Aided Investigative Environment) CD-ROM useful here too, since it's also a live CD-ROM under active development with forensic tools useful in these situations. There is not (yet) a Debian-based tool such as this, nor an easy way to build the CD-ROM using your own selection of Debian packages and mkinitrd-cd (so you'll have to read the documentation provided with it to make your own CD-ROMs).

If you really want to fix the compromise quickly, you should remove the compromised host from your network and re-install the operating system from scratch. Of course, this may not be effective because you will not learn how the intruder got root in the first place. For that case, you must check everything: firewall, file integrity, log host, log files and so on. For more information on what to do following a break-in, see http://www.cert.org/tech_tips/root_compromise.html or SANS's https://www.sans.org/white-papers/.

Algumas perguntas freqüentes de como lidar com um sistema Debian GNU/Linux estão disponíveis em “Meu sistema é vulnerável! (Você tem certeza?)”.

Efetuando backup do sistema

Lembre-se que se você tem certeza de que o sistema foi comprometido você não pode confiar no software instalado ou em qualquer informação retornada por ele. Aplicações podem ser alteradas, módulos do kernel podem ser instalados e etc.

[1] Se você for aventureiro, você pode efetuar o logon no sistema e salvar as informações de todos os processos em execução (várias dessas informações estão em /proc/nnn/). É possível pegar todo código executável da memória, mesmo se o invasor tiver excluído os arquivos executáveis do disco. Então puxe o cabo de força.

[2] >In fact, this is the tool used to build the CD-ROMs for the http://www.gibraltar.at/ project (a firewall on a live CD-ROM based on the Debian distribution).

A melhor coisa a se fazer é uma cópia de backup completa do sistema de arquivo (usando o **dd**) depois de inicializar o sistema de uma mídia segura. Os CDROMs do Debian GNU/Linux podem ser utilizados para isto, já que eles fornecem um shell no console 2 quando a instalação é iniciada (acesse através do Alt +2 e pressione Enter). Do shell, efetue o backup das informações para outro host se possível (talvez um servidor de arquivos de rede através de NFS/FTP). Então qualquer análise da invasão ou reinstalação pode ser feita enquanto o sistema comprometido está off-line.

Se você tiver certeza de que um módulo do kernel com trojan comprometeu o sistema, você pode usar a imagem do kernel do CDROM do Debian no modo *rescue*. Inicie o GNU/Linux no modo *single user* para que nenhum outro processo com trojan seja executado depois do kernel.

Contate seu CERT local

O CERT (Computer and Emergency Response Team) é uma organização que pode te ajudar a recuperar o sistema comprometido. Existem CERTs espalhados por todo o mundo [3] e você deve contatar seu CERT local caso ocorra algum incidente de segurança que comprometa seu sistema. As pessoas do CERT local são orientadas à ajudá-los.

Fornecer informações sobre os incidentes de segurança para o CERT local (ou o centro de coordenação do CERT), mesmo que você não precise de assistência, pode ajudar os outros a determinar se uma vulnerabilidade está disseminada na Internet e indicar que novas ferramentas de combate ao worm estão sendo utilizadas. Estas informações são usadas para fornecer à comunidade da Internet alertas sobre as http://www.cert.org/current/, e para publicar http://www.cert.org/incident_notes/ e até mesmo http://www.cert.org/advisories/. Para informações mais detalhadas de como (e porquê) relatar um incidente leia o http://www.cert.org/tech_tips/incident_reporting.html.

Você pode usar mecanismos menos formais se precisar de ajuda na recuperação de um sistema comprometido ou quiser discutir informações do incidente. Estes mecanismos incluem a http://marc.theaimsgroup.com/?l=incidents e a http://marc.theaimsgroup.com/?l=intrusions.

Análise forense

If you wish to gather more information, the tct (The Coroner's Toolkit from Dan Farmer and Wietse Venema) package contains utilities which perform a *post mortem* analysis of a system. tct allows the user to collect information about deleted files, running processes and more. See the included documentation for more information. These same utilities and some others can be found in http://www.sleuthkit.org/ by Brian Carrier, which provides a web front-end for forensic analysis of disk images. In Debian you can find both sleuthkit (the tools) and autopsy (the graphical front-end).

Também, lembre-se que a análise forense deve ser feita sempre na cópia de backup dos dados, *nunca* nos dados originais, em caso dos dados serem alterados durante a análise e as evidências serem perdidas.

You will find more information on forensic analysis in Dan Farmer's and Wietse Venema's http://www.porcupine.org/forensics/forensic-discovery/ book (available online), as well as in their http://www.porcupine.org/forensics/column.html and their http://www.porcupine.org/forensics/handouts.html. Brian Carrier's newsletter http://www.sleuthkit.org/informer/index.php is also a very good resource on forensic analysis tips. Finally, the http://www.honeynet.org/misc/chall.html are an excellent way to hone your forensic analysis skills as they include real attacks against honeypot systems and provide challenges

[3] Esta é a lista de alguns CERTS, para uma lista completa veja o http://www.first.org/about/organization/teams/index.html (FIRST significa Forum of Incident Response and Security Teams): http://www.auscert.org.au (Austrália), http://www.unam-cert.unam.mx/ (México) http://www.cert.funet.fi (Finlândia), http://www.dfn-cert.de (Alemanha), http://cert.uni-stuttgart.de/ (Alemanha), http://idea.sec.dsi.unim.it (Itália), http://www.jpcert.or.jp/ (Japão), http://cert.uninett.no (Noruega), http://www.cert.hr (Croácia) http://www.cert.pl (Polônia), http://www.cert.ru (Rússia), http://www.arnes.si/si-cert/ (Eslovênia) http://www.rediris.es/cert/ (Espanha), http://www.switch.ch/cert/ (Suiça), http://www.cert.org.tw (Taiwan), e http://www.cert.org (US).

that vary from forensic analysis of disks to firewall logs and packet captures. For information about available forensics packages in Debian visit https://salsa.debian.org and search for *forensic*.

FIXME: This paragraph will hopefully provide more information about forensics in a Debian system in the coming future.

FIXME: talk on how to do a debsums on a stable system with the MD5sums on CD and with the recovered file system restored on a separate partition.

FIXME add pointers to forensic analysis papers (like the Honeynet's reverse challenge or http://staff.washington.edu/dittrich/.

Analysis of malware

Some other tools that can be used for forensic analysis provided in the Debian distribution are: strace and ltrace

Any of these packages can be used to analyze rogue binaries (such as back doors), in order to determine how they work and what they do to the system. Some other common tools include **ldd** (in libc6), **strings** and **objdump** (both in binutils).

If you try to do forensic analysis with back doors or suspected binaries retrieved from compromised systems, you should do so in a secure environment (for example in a bochs or xen image or a **chroot**'ed environment using a user with low privileges[4]). Otherwise your own system can be back doored/r00ted too!

If you are interested in malware analysis then you should read the http://www.porcupine.org/forensics/forensic-discovery/chapter6.html chapter of Dan Farmer's and Wietse Venema's forensics book.

[4] >Be *very* careful if using chroots, since if the binary uses a kernel-level exploit to increase its privileges it might still be able to infect your system

Capítulo 12. Questões feitas com freqüência (FAQ)

Este capítulo introduz algumas das questões mais freqüentes da lista Debian security. Você deverá lê-las antes de postar lá ou senão as pessoas lhe dirão RTFM.

Tornando o sistema operacional Debian mais seguro

A Debian é mais segura que X?

Um sistema é tão seguro quanto um administrador é capaz de faze-lo. A instalação padrão dos serviços da Debian tenta ser *secura*, mas pode não ser paranóica como outros sistemas operacionais que instalam todos os serviços *desativados por padrão*. Em qualquer caso, o administrador de sistemas precisa adaptar a segurança do sistema a sua política de segurança local.

Para uma coleção de dados envolvendo vulnerabilidades de segurança de muitos sistemas operacionais, veja http://securityfocus.com/vulns/stats.shtml. Estes dados são úteis? O site lista diversos fatores a considerar quando estiver interpretando dados, e alerta que os dados não podem ser usados para comparar vulnerabilidades de um sistema operacional versus outro. [1] Também, tenha em mente que algumas das vulnerabilidades reportadas via bugs com relação a Debian, se aplicam somente ao repositório *unstable* (área de desenvolvimento).

A Debian é mais segura que as outras distribuições Linux (tal como Red Hat, SuSE...)?

Realmente não existem muitas diferenças entre as distribuições Linux, com exceção da instalação básica e do sistema de gerenciamento de pacotes. A maioria das distribuições compartilham muitos dos aplicativos, com a diferença básica nas versões em que estes aplicativos são oferecidos com o lançamento da distribuição estável. Por exemplo, o kernel, Bind, Apache, OpenSSH, XFree, gcc, zlib, etc. são todos idênticos entre as distribuições de Linux.

Por exemplo, a Red Hat foi infeliz e ofereceu quando 1.2.3 era a atual, que em seguida foram encontrados problemas de segurança. Na Debian, por outro lado, foi sortuda e forneceu 1.2.4 que já possui a correção da falha. Este foi o caso no grande problema do http://www.cert.org/advisories/CA-2000-17.html diversos anos atrás.

Existe muita colaboração entre os respectivos times de segurança das maiores distribuições Linux. Atualizações de segurança conhecidas são raramente, se existirem, deixadas de lado por desenvolvedores de uma distribuição. O conhecimento de uma vulnerabilidade de segurança nunca é mantida isolada do conhecimento de desenvolvedores de outra distribuição, pois as correções são normalmente coordenadas com o autor ou através do http://www.cert.org. Como um resultado, as atualizações necessárias de segurança são geralmente lançadas ao mesmo tempo e a segurança relativa de diferentes distribuições são bem parecidas.

Uma das principais vantagens da Debian com relação a segurança é a facilidade de atualizações do sistema através do uso do **apt**. Aqui existem muitos outros aspectos da segurança na Debian a serem considerados:

[1] Neste exemplo, baseado nos dados da Securityfocus, pode ser visto que o Windows NT é mais seguro que o Linux, o que é uma afirmação questionável. Apesar de tudo, as distribuições do Linux geralmente oferecem mais aplicações comparadas ao Windows NT da Microsoft. Estas situações de *contagem de vulnerabilidades* são melhor descritas em http://www.dwheeler.com/oss_fs_why.html#security por David A. Wheeler

- A Debian fornece mais ferramentas de segurança que outras distribuições, veja Capítulo 8, *Ferramentas de segurança no Debian*.
- A instalação padrão da Debian é pequena (menos funcionalidades), e assim mais segura. Outras distribuições, em nome da funcionalidade, tem a tendência de instalarem diversos serviços por padrão e algumas vezes não estão corretamente configurados (lembre-se dos http://www.sans.org/y2k/lion.htm). A instalação da Debian não é limitada como o OpenBSD (não existem daemons ativos por padrão), mas tem um bom compromisso. [2]
- A Debian documenta as melhores práticas de segurança em documentos como este.

Existem muitas falhas no sistema de tratamento de falhas da Debian. Isto significa que é muito vulnerável?

A distribuição Debian conta com um número grande e crescente de pacotes de software, provavelmente mais do que os fornecidos por muitos sistemas operacionais proprietários. Quanto mais pacotes instalados, maior o potencial de falhas de segurança em um determinado sistema.

Mais e mais pessoas estão examinando o código fonte por problemas. Existem muitos alertas relacionados com a auditoria de código fonte dos maiores componentes de software incluídos na Debian. Desta forma, tais auditorias de software mostram brechas de segurança, elas são corrigidas e um aviso é enviado para listas tal como Bugtraq.

Falhas que estão presentes na distribuição Debian normalmente também afetam outros distribuidores e vendedores. Verifique a seção "Específico da Debian: yes/no" no topo de cada aviso de segurança (DSA).

A Debian possui qualquer certificação relacionada a segurança?

Resposta curta: não.

Resposta longa: certificação custa dinheiro (especialmente se for uma certificação de segurança *séria*), ninguém dedicou seus recursos para para certificar a Debian GNU/Linux em qualquer nível de, por exemplo, http://niap.nist.gov/cc-scheme/st/. Se estiver interessado em ter uma distribuição de GNU/Linux seguramente certificada, tente fornecer os recursos necessários para tornar isto possível.

There are currently at least two linux distributions certified at different http://en.wikipedia.org/wiki/Evaluation_Assurance_Level levels. Notice that some of the CC tests are being integrated into the http://ltp.sourceforge.net which is available in Debian in the ltp.

Existe algum programa de fortalecimento para a Debian?

Yes. http://bastille-linux.sourceforge.net/, originally oriented toward other Linux distributions (Red Hat and Mandrake), it currently works also for Debian. Steps are being taken to integrate the changes made to the upstream version into the Debian package, named bastille.

Algumas pessoas, no entanto, acreditam que uma ferramenta de fortalecimento não elimina a necessidade de se ter uma boa administração.

Eu desejo executar o serviço XYZ, qual eu devo escolher?

Um dos grandes potenciais da Debian é a grande variedade de escolhas disponíveis entre pacotes que oferecem a mesma funcionalidade (servidores de DNS, servidores de e-mail, servidores ftp, servidores web, etc.). Isto pode confundir o administrador novato ao tentar determinar que pacote é o mais

[2] Sem mencionar o fato que algumas distribuições, tal como a Red Hat ou Mandrake, também estão permitindo que o usuário selecione *perfis de segurança* ou usando assistentes para ajudar na configuração de *firewalls pessoais*.

adequado para você. O melhor para uma determinada situação depende de um balanceamento entre suas características e necessidades de segurança. Aqui estão algumas questões que devem ser feitas a você mesmo quando decidir entre pacotes parecidos:

- Existem um maintainer do código fonte do programa? Quando foi o último lançamento?
- O pacote está maduro? o número de versão realmente *não* mostra sua maturidade. Tente analisar o histórico de atualizações do software.
- Este programa é atormentado por falhas? Tem avisos de segurança relacionados a ele?
- Este programa oferece todas as funcionalidades que precisa? ele oferece mais do que você realmente precisa?

Como eu posso tornar o serviço XYZ mais seguro na Debian?

Você encontrará informações neste documento sobre como tornar alguns serviços (FTP, Bind) mais seguros na Debian GNU/Linux. Para serviços não cobertos aqui, verifique a documentação do programa, ou informações gerais sobre o Linux. Muitas das regras de segurança para sistemas Unix também se aplicam a Debian. Na maioria dos casos, o método para tornar um serviço X mais seguro na Debian é parecido com torná-lo mais seguro em qualquer outra distribuição de Linux (ou Unix, nesta importância).

Como posso remover todos os banners de serviços?

Se não gosta que os usuários que se conectam ao seu serviço de POP3 recebam informações sobre seu sistema (por exemplo), você pode querer remover (ou alterar) o banner que este serviço mostra para os usuários. [3] Fazer isto depende do programa que está executando para um determinado serviço. Por exemplo, no **postfix**, você poderá ajustar o banner SMTP no arquivo `/etc/postfix/main.cf`:

```
smtpd_banner = $myhostname ESMTP $mail_name (Debian/GNU)
```

Other software is not as easy to change. ssh will need to be recompiled in order to change the version that it prints. Take care not to remove the first part (`SSH-2.0`) of the banner, which clients use to identify which protocol(s) is supported by your package.

Todos os pacotes da Debian são seguros?

O time de segurança da Debian não tem a possibilidade de analisar todos os pacotes incluídos na Debian procurando por vulnerabilidades de segurança em potencial, pois não existem recursos para auditar o código fonte de todo o projeto. No entanto, a Debian se beneficia da auditoria de código fonte feita por desenvolvedores que criam o programa.

Como um fato de importância, um desenvolvedor da Debian pode distribuir um Trojan em um pacote e não existe possibilidade de verificar isto. Até mesmo se for introduzido na estrutura da distribuição, seria impossível identificar todas as situações onde o trojan seria executado. Este é o motivo porque a Debian vem com a cláusula de licença *"sem garantias"*.

No entanto, os usuários da Debian podem ter confiança no fato que que código estável tem uma audiência ampla e a maioria dos problemas foram descobertos durante o uso. A instalação de versões não testadas de programas em sistemas críticos é algo não recomendado (se não puder fornecer a auditoria de código necessária). Em qualquer caso, se for descoberta uma vulnerabilidade de segurança introduzida na distribuição, o processo usado para incluir pacote (usando assinaturas digitais) se certifica que o problema pode ser rastreado até o desenvolvedor. O projeto Debian não tem examinado isto levemente.

[3] Note que isto é "segurança pela obscuridade" e provavelmente este esforço não valerá a pena em longo termo.

Porque alguns arquivos de logs/configuração tem permissão de leitura para qualquer um, isto não é inseguro?

É claro, você pode alterar as permissões padrões da Debian em seu sistema. A política atual relacionada com arquivos de log e configuração é que eles sejam lidos por todos *a não ser* que eles contenham informações sensíveis.

Tenha cuidado se fizer estas alterações pois:

- Alguns processos podem não ser capazes de gravar arquivos de log se restringir suas permissões.
- Alguns aplicativos podem deixar de funcionar se o arquivo de configuração que eles dependem não puder ser lido. Por exemplo, se você remover a permissão de leitura para todos do `/etc/samba/smb.conf`, o **smbclient** deixará de funcionar se for executado por um usuário normal.

FIXME: Verificar se isto está escrito na Política. Alguns pacotes (i.e. daemons de ftp) parecem forçar permissões diferentes.

Porque o /root/ (ou UsuarioX) tem permissões 755?

Como fato de importância, as mesmas questões são válidas para qualquer outro usuário. Como a instalação da Debian não coloca *qualquer* arquivo sob aquele diretório, não existe informações sensíveis a serem protegidas lá. Se você sentir que estas permissões são muito largas para seu sistema, considere altera-las para 750. Para os usuários, leia “Limitando acesso a outras informações de usuários”.

This Debian security mailing list http://lists.debian.org/debian-devel/2000/11/msg00783.html has more on this issue.

Após instalar o grsec/firewall, comecei a receber muitas mensagens de console! como removê-las?

Se estiver recebendo mensagens de console e configurou o `/etc/syslog.conf` para redirecioná-las ou para arquivos ou para um TTY especial, você pode ver mensagens sendo direcionadas para a console.

O nível de registro padrão do console para qualquer kernel é 7, o que significa que qualquer mensagem que tem prioridade menor aparecerá no console. Normalmente, os firewalls (a regra LOG) e algumas outras ferramentas de segurança registram eventos em uma prioridade menor que esta, e assim, são enviadas diretamente para a console.

To reduce messages sent to the console, you can use **dmesg** (`-n` option, see dmseg(8)), which examines and *controls* the kernel ring buffer. To fix this after the next reboot, change `/etc/init.d/klogd` from:

```
KLOGD=""
```

para:

```
KLOGD="-c 4"
```

Use um número menor para `-c` se estiver ainda vendo as mensagens. Uma descrição dos diferentes níveis de logs podem ser encontrados no arquivo `/usr/include/sys/syslog.h`:

```
#define LOG_EMERG       0       /* o sistema está inutilizável */
```

```
#define LOG_ALERT       1       /* uma ação deve ser tomada imediatamente */
#define LOG_CRIT        2       /* condições críticas */
#define LOG_ERR         3       /* condições de erro */
#define LOG_WARNING     4       /* condições de alerta */
#define LOG_NOTICE      5       /* condição normal mas significante */
#define LOG_INFO        6       /* informativas */
#define LOG_DEBUG       7       /* mensagens a nível de depuração */
```

Usuários e grupos do sistema operacional

Todos os usuários do sistema são necessários?

Sim e não. A Debian vem com alguns usuários pré-definidos (identificação de usuários (UID) < 99 como descritos na http://www.debian.org/doc/debian-policy/ ou `/usr/share/doc/base-passwd/README`) para facilitar a instalação de alguns serviços que requerem que sejam executados sob um usuário/UID apropriado. Se não tem a intenção de instalar novos serviços, você pode seguramente remover estes usuários que não são donos de qualquer arquivo em seu sistema e não executam qualquer serviço. Em qualquer caso, o comportamento padrão é que UIDs de 0 a 99 são reservadas para a Debian, e UIDs de 100 a 999 são criados por pacotes na instalação (e apagados quando o pacote e suas configurações são removidos do sistema).

To easily find users who don't own any files, execute the following command[4] (run it as root, since a common user might not have enough permissions to go through some sensitive directories):

```
cut -f 1 -d : /etc/passwd | \
while read i; do find / -user "$i" | grep -q . || echo "$i"; done
```

These users are provided by base-passwd. Look in its documentation for more information on how these users are handled in Debian. The list of default users (with a corresponding group) follows:

- root: O root é (tipicamente) o superusuário.
- daemon: Alguns daemons não privilegiados que precisam gravar em arquivos no disco são executados como daemon.daemon (e.g., **portmap**, **atd**, provavelmente outros). Os daemons que não precisam ser donos de quaisquer arquivos são executados sob nobody.nogroup e daemons mais complexos ou com segurança em mente são executados como usuários dedicados. O usuário do daemon é prático para daemons instalados localmente.
- bin: mantido por razões históricas.
- sys: mesmo que bin. No entanto. /dev/vcs* e `/var/spool/cups` tem como donos o grupo sys.
- sync: O interpretador de comandos do usuário sync é `/bin/sync`. Assim se sua senha for ajustada para algo fácil de adivinhar (tal como ""), qualquer um pode fazer sync no sistema pela console, até mesmo se não possuir uma conta.
- games: Muitos jogos são SETGID para games assim eles podem gravar seus arquivos de pontuações. Isto é explicado na policy.
- man: O programa man (algumas vezes) é executado como usuário man, assim ele poderá gravar páginas de manuais em `/var/cache/man`
- lp: Usado por daemons de impressão.

[4] Be careful, as this will traverse your whole system. If you have a lot of disk and partitions you might want to reduce it in scope.

- mail: Caixas de correios em `/var/mail` tem como dono o grupo mail, como explicado pela policy. O usuário e grupo também são usados para outros propósitos por vários MTA's.

- news: Vários servidores de notícias e outros programas associados (tal como o **suck**) utilizam usuário e grupo news de várias formas. Os arquivos no spool de notícias tem freqüentemente como dono o usuário e grupo news. Os programas tais como **inews** que são usados para postar notícias tipicamente usam SETGID para o grupo news.

- uucp: O usuário e grupo uucp são usados pelo subsistema UUCP. Ele é dono do spool e arquivos de configuração. Usuários no grupo uucp podem executar o uucico.

- proxy: Assim como o daemon, este usuário e grupo são usados por alguns daemons (especificamente, daemons de proxy) que precisam de identificação de usuários dedicadas para ser dono de arquivos. Por exemplo, o grupo proxy é usado pelo **pdnsd** e **squid** para serem executados como o usuário proxy.

- majordom: **Majordomo** tem uma UID estaticamente alocada em sistemas Debian por razões históricas. Ele não é instalado em novos sistemas.

- postgres: Os bancos de dados do **Postgresql** tem como dono este usuário e grupo. Todos os arquivos sob `/var/lib/postgresql` tem como dono este usuário para forçar segurança de forma apropriada.

- www-data: Alguns servidores web são executados sob www-data. O conteúdo web *não* deve ter como dono este usuário, ou um servidor web comprometido poderia ser capaz de regravar um site de internet. Dados gravados por servidores web, incluindo arquivos de logs, terão que ter como dono www-data.

- backup: Assim as responsabilidades de backup/restauração podem ser localmente delegadas para alguém sem permissões completas de usuário root.

- operator: O operador é historicamente (e praticamente) a única conta de "usuário" que pode efetuar login remotamente, e não depende do NIS/NFS.

- list: Os arquivos de listas de discussões e dados tem como dono este usuário e grupo. Alguns programas de listas de discussões podem ser executadas também sobe este usuário.

- irc: Usado por daemons de irc. É necessário um usuário alocado estaticamente somente por causa de um bug no **ircd**, que faz SETUID()s de si mesmo para a UID especificada na inicialização.

- gnats.

- nobody, nogroup: Daemons que não tem necessidade de serem donos de quaisquer arquivos são executados sob o usuário nobody e grupo nogroup. Assim, nenhum arquivo existente no sistema devem ter como donos este usuário ou grupo.

Outros grupos que não tem um usuário associado:

- adm: O grupo adm é usado para tarefas de monitoramento do sistema. Os membros deste grupo podem ler a maioria dos arquivos de log em `/var/log` e podem usar o xconsole. Historicamente, o `/var/log` foi `/usr/adm` (e depois `/var/adm`), isto explica o nome do grupo.

- tty: Os dispositivos TTY tem como dono este grupo. Eles são usados pelas ferramentas write e wall para permitir escrever para pessoas conectadas em outras TTYs.

- disk: Acesso direto a disco. Muito equivalente ao acesso root.

- kmem: /dev/kmem e arquivos similares são lidos por este grupo. Isto é mais uma relíquia do BSD, mas alguns programas que precisam de acesso de leitura direto a memória do sistema podem fazer SETGID para o grupo kmem.

- dialout: Acesso direto e completo a portas seriais. Membros deste grupo podem reconfigurar o modem, discar para qualquer lugar, etc.
- dip: O nome do grupo vem de "Dial-up IP", e membros que pertencem ao grupo dip podem usar ferramentas como o **ppp**, **dip**, **wvdial**, etc. para realizar uma conexão. Os usuários neste grupo não podem reconfigurar o modem, mas podem executar programas para fazerem uso dele.
- fax: Permite que membros usem programas de fax para ler/enviar faxes.
- voice: Voicemail, útil para sistemas que usam modens como secretárias eletrônicas.
- cdrom: Este grupo pode ser usado localmente para dar ao grupo de usuários acesso a unidade de CDROM.
- floppy: Este grupo pode ser usado localmente par dar a um grupo de usuários acesso a unidade de disquetes.
- tape: Este grupo pode ser usado localmente para dar a um grupo de usuários acesso a uma unidade de fita.
- sudo: Membros dentro deste grupo não precisam digitar sua senha quando estiverem fazendo o uso do **sudo**. Veja `/usr/share/doc/sudo/OPTIONS`.
- audio: Este grupo pode ser usado localmente para dar a um grupo de usuários acesso a um dispositivo de audio.
- src: Este grupo é dono de código fonte, incluindo arquivos em `/usr/src`. Ele pode ser usado para dar a um usuário a habilidade de gerenciar código fonte do sistema.
- shadow: O arquivo `/etc/shadow` é lido por este grupo. Alguns programas que precisam ser capazes de acessar o arquivo tem SETGID ajustados para shadow.
- utmp: Este grupo pode gravar para o arquivo `/var/run/utmp` e similares. Programas que precisam se capazes de gravar para ele usam SETGID para utmp.
- video: Este grupo é usado localmente para dar a um conjunto de usuários permissões de acesso a dispositivos de vídeo.
- staff: Permite que usuários adicionem modificações locais ao sistema (`/usr/local`, `/home`) sem necessidade de privilégios de usuário root. Compare com o grupo "adm", que é mais relacionado a segurança/monitoramento.
- users: Enquanto usuários de sistemas Debian usam seus grupos privados de sistema por padrão (cada usuário tem seu próprio grupo), alguns preferem usar um grupo de sistema mais tradicional, no qual cada usuário é membro de seu grupo.

I removed a system user! How can I recover?

If you have removed a system user and have not made a backup of your `password` and `group` files you can try recovering from this issue using **update-passwd** (see update-passwd(8)).

Quais são as diferenças entre os grupos adm e staff?

Componentes do grupo "adm" são geralmente administradores e neste grupo as permissões os permitem ler arquivos de log sem utilizar **su**. O grupo "staff" são geralmente administradores junior e de suporte, permitindo que trabalhem em `/usr/local` e criarem diretórios em `/home`.

Porque existe um novo grupo quando adiciono um novo usuário? (ou porque a Debian cria um novo grupo para cada usuário?)

O comportamento padrão na Debian é que cada usuário tem seu próprio e privado grupo. O esquema tradicional do UN*X coloca todos os usuários no grupo *users*. Grupos adicionais foram criados e usados para restringir o acesso a arquivos compartilhados associados com diferentes diretórios de projetos. O gerenciamento de arquivos se torna difícil quando apenas um usuário trabalha em múltiplos projetos, porque quando alguém cria um arquivo, ele é associado com o grupo primário do grupo que ele pertence (e.g. "users").

O método da Debian resolve este problema associando a cada usuário seu próprio grupo; assim com a máscara apropriada (0002) e o bit SETGID ajustado em um diretório determinado de projetos, o grupo correto é automaticamente designado para arquivos criados naquele diretório. Isto facilita a vida de pessoas que trabalham em múltiplos projetos, porque elas não terão que alterar os grupos e umasks quando estiverem trabalhando em arquivos compartilhados.

Você pode, no entanto, alterar este comportamento modificando o `/etc/adduser.conf`. Altere a variável *USERGROUPS* para "no", assim um novo grupo não será criado quando o novo usuário for criado. Também, altere *USERS_GID* para a identificação de grupo a que os usuários pertencem.

Questões relacionadas a serviços e portas abertas

Porque todos os serviços são ativados durante a instalação?

Esta é simplesmente uma aproximação do problema de sendo, de um lado, consciente de segurança e por outro lado amigável ao usuário. De forma contrária a OpenBSD, que desativa todos os serviços a não ser que sejam ativados pelo administrador, a Debian GNU/Linux ativa todos os serviços instalados a não ser que sejam desativados (veja “Desabilitando daemons de serviço” para mais informações). Afinal, você instalou o serviço, não foi?

Existem muitas discussões nas listas de discussões da Debian (ambas na debian-devel e na debian-security) com relação a qual é a melhor estratégia para a instalação padrão. No entanto, no momento em que isto foi escrito (Março de 2002), ainda não existia um consenso.

Can I remove inetd?

Inetd is not easy to remove since netbase depends on the package that provides it (netkit-inetd). If you want to remove it, you can either disable it (see “Desabilitando daemons de serviço”) or remove the package by using the equivs package.

Porque eu tenho a porta 111 aberta?

A porta 111 é usada pelo portmapper sunrpc e é instalada por padrão como parte do sistema de instalação básico da Debian, pois não existe a necessidade de saber quando o programa do usuário precisa do RPC para funcionar adequadamente. Em qualquer caso, ele é mais usado pelo NFS. Se não precisar dele, remova-o como explicado na seção “Tornando serviços RPC mais seguros”.

In versions of the portmap package later than 5-5 you can actually have the portmapper installed but listening only on localhost (by modifying `/etc/default/portmap`)

Para que a porta 113 (identd) é usada?

O serviço ident é um serviço de autenticação que identifica o dono de uma conexão TCP/IP para o servidor remoto que está aceitando a conexão. Tipicamente, quando um usuário se conecta ao servidor remoto,

o **inetd** do sistema remoto envia uma requisição à porta 113 para procurar informações sobre o dono. É freqüentemente usada em servidores de e-mails, FTP e IRC, e também podem ser usadas para descobrir que usuário em seu sistema local está atacando um sistema remoto.

There has been extensive discussion on the security of **identd** (See http://lists.debian.org/debian-security/2001/08/msg00297.html). In general, **identd** is more helpful on a multi-user system than on a single user workstation. If you don't have a use for it, disable it, so that you are not leaving a service open to the outside world. If you decide to firewall the identd port, *please* use a reject policy and not a deny policy, otherwise a connection to a server utilizing **identd** will hang until a timeout expires (see http://logi.cc/linux/reject_or_deny.php3).

Tenho serviços usando a porta 1 e 6, o que são e como posso removê-las?

Se executar o comando `netstat -an` e receber como retorno:

```
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
PID/Program name
raw        0      0 0.0.0.0:1               0.0.0.0:*               7
-
raw        0      0 0.0.0.0:6               0.0.0.0:*               7
-
```

You are *not* seeing processes listening on TCP/UDP port 1 and 6. In fact, you are seeing a process listening on a *raw* socket for protocols 1 (ICMP) and 6 (TCP). Such behavior is common to both legitimate software like intrustion detection systems, such as iplogger and portsentry, but some trojans have also been known yo use them. If you have the mentioned packages simply remove them to close the port. If you do not, try netstat's `-p` (process) option to see which process is running these listeners.

Encontrei a porta XYZ aberta, posso fechá-la?

Sim, com certeza. As portas que está deixando abertas devem aderir a política individual do seu site com relação a serviços públicos disponíveis para outras redes. Verifique se estão sendo abertas pelo **inetd** (veja “Desabilitando o **inetd** ou seus serviços”) ou instalando pacotes individuais e tome as medidas apropriadas (i.e, configure o inetd, remova o pacote, evite executá-lo na inicialização).

Removendo serviços do `/etc/services` ajudará a tornar minha máquina mais segura?

Não o `/etc/services` somente oferece o mapeamento entre um nome virtual e um número dado de porta. A remoção de nomes deste arquivo (geralmente) não evitará que os serviços sejam iniciados. Alguns daemons podem não ser executados se o `/etc/services` for modificado mas isto não é a norma. Para desativar apropriadamente o serviço, veja “Desabilitando daemons de serviço”.

Assuntos comuns relacionados a segurança

Perdi minha senha e não posso acessar o sistema!

Os passos que precisa fazer para se recuperar disto depende se aplicou ou não os procedimentos necessários para limitar o acesso ao **lilo** e da BIOS do seu sistema.

Se limitou ambos, precisará desativar a configuração de BIOS que somente lhe permite inicializar através do disco rígido antes de prosseguir. Se tiver também perdido a senha da sua BIOS, você terá que resetar a sua BIOS abrindo o computador e removendo manualmente a bateria que mantém os dados da BIOS>

Assim que permitir a inicialização através da unidade de CD-ROM ou ativação da unidade de disquete, faça o seguinte:

- Inicialize através de um disquete de recuperação e inicie o kernel
- Vá até o console virtual (Alt+F2)
- Monte o disco rígido onde o sistema de arquivos raíz (/) está
- Edite o arquivo `/etc/shadow` (o disquete de recuperação da Debian 2.2 vem com o editor **ae** e a Debian 3.0 vem com o **nano-tiny** que é similar ao **vi**) e altere a linha:

  ```
  root:asdfjgl29gl0341274075:XXXX:X:XXXX:X::: (X=um número qualquer)
  ```

  para:

  ```
  root::XXXX:X:XXXX:X:::
  ```

Isto removerá a senha de root perdida, contida no primeiro campo separado por dois pontos após o nome do usuário. Salve o arquivo, reinicie o sistema e faça login como usuário root usando uma senha em branco. Lembre-se de adicionar uma nova senha. Isto funcionará a menos que tenha configurado o sistema de forma mais restrita, ou seja, não permitindo que usuários utilizem senhas em branco ou não permitindo o login do usuário root através do console.

Se adicionou estas características, você precisará entrar em modo monousuário. Se o LILO foi restringido, será necessário re-executar o **lilo** após alterar a senha de root acima. Este truque é necessário pois seu `/etc/lilo.conf` precisa ser mexido devido ao sistema de arquivos raíz (/) ser um disco ram e não um disco rígido real.

Assim que a restrição do LILO for removida, tente o seguinte:

- Pressione as teclas Alt, shift e Control antes do sistema terminar o processo de inicialização, assim você terá acesso ao aviso de comandos do LILO.
- Digite `linux single`, `linux init=/bin/sh` ou `linux 1` na linha de comandos.
- Isto lhe dará um aviso de comandos do shell em modo monousuário (ele perguntará por uma senha, mas você já a conhece)
- Remonte sua partição raíz (/) usando o comando mount.

  ```
  # mount -o remount,rw /
  ```

- Altere a senha do usuário root com o comando **passwd** (como você é o superusuário, o sistema não perguntará a senha anterior).

Como posso configurar um serviço para meus usuários sem lhes dar uma conta de acesso ao shell?

For example, if you want to set up a POP service, you don't need to set up a user account for each user accessing it. It's best to set up directory-based authentication through an external service (like Radius, LDAP or an SQL database). Just install the appropriate PAM library (libpam-radius-auth, libpam-ldap, libpam-pgsql or libpam-mysql), read the documentation (for starters, see “Autenticação do Usuário:

PAM") and configure the PAM-enabled service to use the back end you have chosen. This is done by editing the files under `/etc/pam.d/` for your service and modifying the

```
auth   required    pam_unix_auth.so shadow nullok use_first_pass
```

para, por exemplo, ldap:

```
auth   required    pam_ldap.so
```

No caso de diretórios LDAP, alguns serviços oferecem esquemas LDAP que devem ser incluídos em seu diretório e são necessários para a utilização de autenticação LDAP. Se estiver usando um banco de dados relacional, uma dica útil é usar a cláusula *where* quando estiver configurando os módulos do PAM. Por exemplo, se tiver um banco de dados com os seguintes atributos na tabela:

```
(user_id, user_name, realname, shell, password, UID, GID, homedir, sys, pop, ima
```

Tornando os serviços campos de atributos boleanos, você poderá usa-los para permitir ou negar acesso a diferentes serviços apenas inserindo as linhas apropriadas nos seguintes arquivos:

- `/etc/pam.d/imap`:`where=imap=1`.
- `/etc/pam.d/qpopper`:`where=pop=1`.
- `/etc/nss-mysql*.conf`:`users.where_clause = user.sys = 1;`.
- `/etc/proftpd.conf`: `SQLWhereClause "ftp=1"`.

Meu sistema é vulnerável! (Você tem certeza?)

O scanner de vulnerabilidade X diz que meu sistema Debian é vulnerável!

Muitos scanners de avaliação de vulnerabilidades indicarão falso positivos quando forem usados em sistemas Debian, pois podem somente usar checagem de versões para determinar se uma determinada versão de pacote é vulnerável, mas realmente não testam a vulnerabilidade de segurança propriamente dita. Pois a Debian não muda os números de versões quando corrige um pacote (muitas vezes a correção feita em versões novas são reproduzidas nas atuais), algumas ferramentas tendem a achar que um sistema Debian atualizado está vulnerável, quando não está.

Se você acha que o seu sistema está atualizado com patches de segurança, você pode querer usar as referências cruzadas com o banco de dados de vulnerabilidades publicados com os DSAs (veja "Debian Security Advisories") para afastar a possibilidade de falsos positivos, se a ferramenta que estiver usando inclui referências do CVE.

Eu vi um ataque em meus logs de sistema. Meu sistema foi comprometido?

Um traço de ataque nem sempre significa que seu sistema foi comprometido, e você deverá fazer os passos tradicionais para determinar se o sistema está comprometido (veja Capítulo 11, *Depois do comprometimento do sistema (resposta a incidentes)*). Também, note que o fato de ver os ataques nos logs pode significar que seu sistema está vulnerável a ele (um invasor determinado pode ter usado outras vulnerabilidades que não sejam a que você viu, no entanto).

Eu vi algumas linhas estranhas "MARK" em meus logs: Eu fui comprometido?

Você pode achar as seguintes linhas nos seus logs de sistema:

```
Dec 30 07:33:36 debian -- MARK --
Dec 30 07:53:36 debian -- MARK --
Dec 30 08:13:36 debian -- MARK --
```

Isto não indica qualquer tipo de comprometimento e os usuários que estão mudando de versão da Debian devem achar isto estranho. Se o seu sistema não tem uma carga alta (ou muitos serviços ativos), estas linhas devem aparecer entre seus logs. Isto é uma indicação que seu daemon do **syslogd** está sendo executado de forma apropriada. Texto extraído da página de manual syslogd(8):

```
-m intervalo
       O syslogd registra uma marca de horário regularmente. O
       intervalo padrão entre duas linhas -- MARK -- é de 20 minutos.
       Isto pode ser alterado com esta opção.
       O intervalo de zero, desativa totalmente este recurso.
```

Encontrei usuários usando o "su" em meus logs: Eu fui comprometido?

Você pode encontrar linhas em seus logs como:

```
Apr  1 09:25:01 server su[30315]: + ??? root-nobody
Apr  1 09:25:01 server PAM_unix[30315]: (su) session opened for user nobody by (
```

Não se preocupe muito. Verifique para ver se estas mensagens são devido a tarefas do **cron** (normalmente `/etc/cron.daily/find` ou **logrotate**):

```
$ grep 25 /etc/crontab
25 6    * * *   root    test -e /usr/sbin/anacron || run-parts --report
/etc/cron.daily
$ grep nobody /etc/cron.daily/*
find:cd / && updatedb --localuser=nobody 2>/dev/null
```

Encontrei um possível "SYN flooding" em meus logs: Estou sob um ataque?

Se ver linhas como estas em seus logs:

```
May 1 12:35:25 linux kernel: possible SYN flooding on port X. Sending cookies.
May 1 12:36:25 linux kernel: possible SYN flooding on port X. Sending cookies.
May 1 12:37:25 linux kernel: possible SYN flooding on port X. Sending cookies.
May 1 13:43:11 linux kernel: possible SYN flooding on port X. Sending cookies.
```

Verifique se existe um número alto de conexões ao servidor usando o **netstat**, por exemplo:

```
linux:~# netstat -ant | grep SYN_RECV | wc -l
   9000
```

Isto é uma indicação de ataque de negação de serviço (denial of service - DoS) contra a porta X do seu sistema (mais provável contra um serviço público tal como um servidor web ou servidor de e-mails). Você deverá ativar os SynCookies TCP em seu kernel, veja “Configuring syncookies”. Note, no entanto, que um ataque DoS pode sobrecarregar sua rede até mesmo se você puder parar de fazê-lo travar seus sistemas (devido ao número de descritores de arquivos sendo reduzidos, o sistema pode parar de responder até que o tempo limite de algumas conexões se esgote). O único método efetivo de parar este ataque é contactar seu provedor de rede.

Encontrei seções de root estranhas em meus logs: Eu fui comprometido?

Se ver estes tipos de entradas em seu arquivo `/var/log/auth.log`:

```
May 2 11:55:02 linux PAM_unix[1477]: (cron) session closed for user root
May 2 11:55:02 linux PAM_unix[1476]: (cron) session closed for user root
May 2 12:00:01 linux PAM_unix[1536]: (cron) session opened for user root by
(UID=0)
May 2 12:00:02 linux PAM_unix[1536]: (cron) session closed for user root
```

Estas são devido a uma tarefa do **cron** sendo executada (neste exemplo, a cada cinco minutos). Para determinar que programa é responsável por estas tarefas, verifique as tarefas nos diretórios: `/etc/crontab`, `/etc/cron.d`, `/etc/crond.daily` e do root `crontab` sob `/var/spool/cron/crontabs`.

Sofri uma invasão, o que faço?

Existem diversos passos que deve fazer no caso de uma invasão:

- Verifique se o seu sistema está atualizado com as atualizações de segurança de vulnerabilidades publicadas. Se o seu sistema estiver vulnerável, as chances do sistema estar de fato comprometido são maiores. As chances crescem mais se a vulnerabilidade foi conhecida durante algum tempo, pois normalmente existem mais atividades com relação a vulnerabilidades antigas. Aqui está um link para http://www.sans.org/top20/.

- Leia este documento, especialmente a seção Capítulo 11, *Depois do comprometimento do sistema (resposta a incidentes)*

- Peça assistência. Você deverá usar a lista de discussão debian-security para perguntar sobre como recuperar/corrigir seu sistema.

- Notifique seu http://www.cert.org local (caso ele exista, caso contrário você deverá considerar o contato direto com o CERT). Isto pode ou não ajudar você, mas, pelo menos, informará o CERT de ataques que estejam acontecendo. Esta informação é muito valiosa em determinar que ferramentas e ataques estão sendo usados pela comunidade *chapéu preto*.

Como posso rastrear um ataque?

Olhando os logs (caso não tenham sido mexidos) usando sistemas de detecção de intrusão (veja “Configure um sistema de Detecção de Intrusão”), **traceroute**, **whois** e ferramentas parecidas (incluindo análise

forense), você pode ser capaz de detectar um ataque até a sua origem. O método que pode reagir a esta informação depende solenemente de sua política de segurança e o que *você* considera um ataque. Um scan remoto é um ataque? É um teste de vulnerabilidade um ataque?

O programa X na Debian é vulnerável, o que fazer?

Primeiro, leve um momento para se certificar se a vulnerabilidade foi anunciada em listas de discussões de segurança públicas (como a Bugtraq) ou outros fóruns. O time da Debian Security se mantém atualizada com estas listas, assim elas também deverão ter conhecimento do problema. Não faça qualquer outra ações se você ver um anúncio em http://security.debian.org.

Caso nenhuma informação tenha sido publicada, por favor envie um e-mail sobre o(s) pacote(s) afetado(s), assim como uma descrição detalhada da vulnerabilidade (código que comprova isto também é válido) para mailto:team@security.debian.org. Isto lhe colocará em contato com o time de segurança da Debian.

O número de versão de um pacote indica que eu ainda estou usando uma versão vulnerável!

Ao invés de atualizar para uma versão nova, a Debian adapta as correções para a versão que é fornecida com o lançamento estável. A razão disto é para ter certeza que o lançamento estável altere o mínimo possível, assim as coisas não alterarão ou quebrarão de forma inesperada como resultado de uma correção de falha. Você pode verificar se está executando uma versão segura de pacote olhando nos logs de alterações do pacote ou comparando seu número de versão exato (versão do autor - traço- lançamento da Debian) com o número de versão indicado no aviso de segurança da Debian.

Programas específicos

proftpd is vulnerable to a Denial of Service attack.

Adicione `DenyFilter \*.*/` em seu arquivo de configuração, e para mais informações veja http://www.proftpd.org/critbugs.html.

After installing portsentry, there are a lot of ports open.

Este é simplesmente o método como o **portsentry** funciona. Ele abre cerca de vinte portas não usadas para tentar identificar port scans.

Questões relacionadas ao time de segurança da Debian

The security team keeps its list of Frequently Asked Questions at the http://www.debian.org/security/faq. Please refer to that web page for up to date information.

Apêndice A. Histórico de Revisões

Histórico de Revisões

Revisão 3-19.2 Sun May 19 2024 HolgerWansing<hwansing@mailbox.org>

Translation files synchronised with XML sources 3-19

Revisão 3-19.1 Mon May 1 2017 MarcosFouces<marcos.fouces@gmail.com>

Translation files synchronised with XML sources 3-19

Revisão 3-19 April 2017 MarcosFouces<marcos.fouces@gmail.com>

Migrate to Docbook XML.

Build with Publican. No longer use custom Makefile.

Migrate svn repository to git.

Import chinese, italian, spanish, portuguese, japanese, russian, french and german translations to PO format.

Revisão 3-18 February 2015 ThijsKinkhorst<thijs@debian.org>

Clarify FAQ on raw sockets.

Update section 4.5 on GRUB2.

Replace example postrm user removal code with advice to use deluser/delgroup --system

Revisão 3-17 January 2015 ThijsKinkhorst<thijs@debian.org>

Remove mention of MD5 shadow passwords.

Do not recommend dselect for holding packages.

No longer include the Security Team FAQ verbatim, because it duplicates information documented elsewhere and is hence perpetually out of date.

Update section on restart after library upgrades to mention needrestart.

Avoid gender-specific language. Patch by Myriam.

Use LSB headers for firewall script. Patch by Dominic Walden.

Revisão 3-16 January 2013 JavierFernández-Sanguino Peña.<jfs@debian.org>

Indicate that the document is not updated with latest versions.

Update pointers to current location of sources.

Update information on security updates for newer releases.

Point information for Developers to online sources instead of keeping the information in the document, to prevent duplication.

Extend the information regarding securing console access, including limiting the Magic SysRq key.

Update the information related to PAM modules including how to restrict console logins, use cracklib and use the features avialable in /etc/pam.d/login. Remove the references to obsolete variables in /etc/login.defs.

Reference some of the PAM modules available to use double factor authentication, for administrators that want to stop using passwords altogether.

Fix shell script example in Appendix.

Fix reference errors.

Point to the Basille sourceforge project instead of the bastille-unix.org site as it is not responding.

Revisão 3-15 December 2010 JavierFernández-Sanguino Peña<jfs@debian.org>

Change reference to Log Analysis' website as this is no longer available.

Revisão 3-14 March 2009 JavierFernández-Sanguino Peña<jfs@debian.org>

Change the section related to choosing a filesystem: note that ext3 is now the default.

Change the name of the packages related to enigmail to reflect naming changes introduced in Debian.

Revisão 3-13 February 2008 JavierFernández-Sanguino Peña<jfs@debian.org>

Change URLs pointing to Bastille Linux to www.Bastille-UNIX.org since the domain has been http://bastille-linux.sourceforge.net/press-release-newname.html.

Fix pointers to Linux Ramen and Lion worms.
Use linux-image in the examples instead of the (old) kernel-image packages.
Fix typos spotted by Francesco Poli.

Revisão 3-12 August 2007 JavierFernández-Sanguino Peña<jfs@debian.org>

Update the information related to security updates. Drop the text talking about Tiger and include information on the update-notifier and adept tools (for Desktops) as well as debsecan. Also include some pointers to other tools available.
Divide the firewall applications based on target users and add fireflier to the Desktop firewall applications list.
Remove references to libsafe, it's not in the archive any longer (was removed January 2006).
Fix the location of syslog's configuration, thanks to John Talbut.

Revisão 3-11 January 2007 JavierFernández-Sanguino Peña<jfs@debian.org>

Thanks go to Francesco Poli for his extensive review of the document.
Remove most references to the woody release as it is no longer available (in the archive) and security support for it is no longer available.
Describe how to restrict users so that they can only do file transfers.
Added a note regarding the debian-private declasiffication decision.
Updated link of incident handling guides.
Added a note saying that development tools (compilers, etc.) are not installed now in the default 'etch' installation.
Added a note saying that development tools (compilers, etc.) are not installed now in the default 'etch' installation.
Fix references to the master security server.
Add pointers to additional APT-secure documentation.
Improve the description of APT signatures.
Comment out some things which are not yet final related to the mirror's official public keys.
Fixed name of the Debian Testing Security Team.
Remove reference to sarge in an example.
Update the antivirus section, clamav is now available on the release. Also mention the f-prot installer.
Removes all references to freeswan as it is obsolete.
Describe issues related to ruleset changes to the firewall if done remotely and provide some tips (in footnotes).
Update the information related to the IDS installation, mention BASE and the need to setup a logging database.
Rewrite the "running bind as a non-root user" section as this no longer applies to Bind9. Also remove the reference to the init.d script since the changes need to be done through /etc/default.
Remove the obsolete way to setup iptables rulesets as woody is no longer supported.
Revert the advice regarding LOG_UNKFAIL_ENAB it should be set to 'no' (as per default).
Added more information related to updating the system with desktop tools (including update-notifier) and describe aptitude usage to update the system. Also note that dselect is deprecated.
Updated the contents of the FAQ and remove redundant paragraphs.
Review and update the section related to forensic analysis of malware.
Remove or fix some dead links.
Fix many typos and gramatical errors reported by Francesco Poli.

Revisão 3-10 November 2006 JavierFernández-Sanguino Peña<jfs@debian.org>

Provide examples using apt-cache's rdepends as suggested by Ozer Sarilar.
Fix location of Squid's user's manual because of its relocation as notified by Oskar Pearson (its maintainer).
Fix information regarding umask, it's logins.defs (and not limits.conf) where this can be configured for all login connections. Also state what is Debian's default and what would be a more restrictive value for both users and root. Thanks to Reinhard Tartler for spotting the bug.

Revisão 3-9 October 2006 JavierFernández-Sanguino Peña<jfs@debian.org>
Add information on how to track security vulnerabilities and add references to the Debian Testing Security Tracker.
Add more information on the security support for testing.
Fix a large number of typos with a patch provided by Simon Brandmair.
Added section on how to disable root prompt on initramfs provided by Max Attems.
Remove references to queso.
Note that testing is now security-supported in the introduction.

Revisão 3-8 July 2006 JavierFernández-Sanguino Peña<jfs@debian.org>
Rewrote the information on how to setup ssh chroots to clarify the different options available, thank to Bruce Park for bringing up the different mistakes in this appendix.
Fix lsof call as suggested by Christophe Sahut.
Include patches for typo fixes from Uwe Hermann.
Fix typo in reference spotted by Moritz Naumann.

Revisão 3-7 April 2006 JavierFernández-Sanguino Peña<jfs@debian.org>
Add a section on Debian Developer's best practices for security.
Ammended firewall script with comments from WhiteGhost.

Revisão 3-6 March 2006 JavierFernández-Sanguino Peña<jfs@debian.org>
Included a patch from Thomas Sjögren which describes that `noexec` works as expected with "new" kernels, adds information regarding tempfile handling, and some new pointers to external documentation.
Add a pointer to Dan Farmer's and Wietse Venema's forensic discovery web site, as suggested by Freek Dijkstra, and expanded a little bit the forensic analysis section with more pointers.
Fixed URL of Italy's CERT, thanks to Christoph Auer.
Reuse Joey Hess' information at the wiki on secure apt and introduce it in the infrastructure section.
Review sections referring to old versions (woody or potato).
Fix some cosmetic issues with patch from Simon Brandmair.
Included patches from Carlo Perassi: acl patches are obsolete, openwall patches are obsolete too, removed fixme notes about 2.2 and 2.4 series kernels, hap is obsolete (and not present in WNPP), remove references to Immunix (StackGuard is now in Novell's hands), and fix a FIXME about the use of bsign or elfsign.
Updated references to SElinux web pages to point to the Wiki (currently the most up to date source of information).
Include file tags and make a more consistent use of "MD5 sum" with a patch from Jens Seidel.
Patch from Joost van Baal improving the information on the firewall section (pointing to the wiki instead of listing all firewall packages available) (Closes: #339865).
Review the FAQ section on vulnerability stats, thanks to Carlos Galisteo de Cabo for pointing out that it was out of date.
Use the quote from the Social Contract 1.1 instead of 1.0 as suggested by Francesco Poli.

Revisão 3-5 November 2005 JavierFernández-Sanguino Peña<jfs@debian.org>
Note on the SSH section that the chroot will not work if using the nodev option in the partition and point to the latest ssh packages with the chroot patch, thanks to Lutz Broedel for pointing these issues out.
Fix typo spotted by Marcos Roberto Greiner (md5sum should be sha1sum in code snippet).
Included Jens Seidel's patch fixing a number of package names and typos.
Slightly update of the tools section, removed tools no longer available and added some new ones.
Rewrite parts of the section related to where to find this document and what formats are available (the website does provide a PDF version). Also note that copies on other sites and translations might be obsolete (many of the Google hits for the manual in other sites are actually out of date).

Revisão 3-4 August-September 2005 JavierFernández-Sanguino Peña<jfs@debian.org>

Improved the after installation security enhancements related to kernel configuration for network level protection with a sysctl.conf file provided by Will Moy.
Improved the gdm section, thanks to Simon Brandmair.
Typo fixes from Frédéric Bothamy and Simon Brandmair.
Improvements in the after installation sections related to how to generate the MD5 (or SHA-1) sums of binaries for periodic review.
Updated the after installation sections regarding checksecurity configuration (was out of date).

Revisão 3-3	June 2005	JavierFernández-Sanguino Peña<jfs@debian.org>

Added a code snippet to use grep-available to generate the list of packages depending on Perl. As requested in #302470.
Rewrite of the section on network services (which ones are installed and how to disable them).
Added more information to the honeypot deployment section mentioning useful Debian packages.

Revisão 3-2	March 2005	JavierFernández-Sanguino Peña<jfs@debian.org>

Expanded the PAM configuration limits section.
Added information on how to use pam_chroot for openssh (based on pam_chroot's README).
Fixed some minor issues reported by Dan Jacobson.
Updated the kernel patches information partially based on a patch from Carlo Perassi and also by adding deprecation notes and new kernel patches available (adamantix).
Included patch from Simon Brandmair that fixes a sentence related to login failures in terminal.
Added Mozilla/Thunderbird to the valid GPG agents as suggested by Kapolnai Richard.
Expanded the section on security updates mentioning library and kernel updates and how to detect when services need to be restarted.
Rewrote the firewall section, moved the information that applies to woody down and expand the other sections including some information on how to manually set the firewall (with a sample script) and how to test the firewall configuration.
Added some information preparing for the 3.1 release.
Added more detailed information on kernel upgrades, specifically targeted at those that used the old installation system.
Added a small section on the experimental apt 0.6 release which provides package signing checks. Moved old content to the section and also added a pointer to changes made in aptitude.
Typo fixes spotted by Frédéric Bothamy.

Revisão 3-1	January 2005	JavierFernández-Sanguino Peña<jfs@debian.org>

Added clarification to ro /usr with patch from Joost van Baal.
Apply patch from Jens Seidel fixing many typos.
FreeSWAN is dead, long live OpenSWAN.
Added information on restricting access to RPC services (when they cannot be disabled) also included patch provided by Aarre Laakso.
Atualização do script apt-check-sigs do aj.
Aplicação do patch de Carlo Perassi corrigindo URLs.
Aplicação do patch de Davor Ocelic corrigindo muitos erros, enganos, urls, gramática e FIXMEs. Também adicionadas mais informações adicionais a respeito de algumas seções.
Reescrita a seção sobre auditoria do usuário, destacando o uso do script que não tem as mesmas restrições associadas ao histórico do shell.

Revisão 3-0	December 2004	JavierFernández-Sanguino Peña<jfs@debian.org>

Reescrita as informações sobre auditoria de usuário incluindo exemplos de como usar o script.

Revisão 2-99	March 2004	JavierFernández-Sanguino Peña<jfs@debian.org>

Adicionadas informações sobre referências nos DSAs e compatibilidade com o CVE.
Adicionadas informações a respeito do apt 0.6 (apt-secure colocado na experimental)

Corrigida a localização do HOWTO sobre como executar daemons em ambiente chroot como sugerido por Shuying Wang.
Alterada a linha do APACHECTL no exemplo de chroot do Apache (até se não for usado) como sugerido por Leonard Norrgard.
Adicionada uma nota de rodapé a respeito de ataques usando hardlinks caso as partições não fossem configuradas adequadamente.
Adicionada passos faltantes para executar o bind como named como descrito por Jeffrey Prosa.
Adicionada notas sobre o Nessus e Snort desatualizados na woody e disponibilidade de pacotes portados para esta versão.
Adicionado um capítulo a respeito da checagem de integridade periódica.
Esclarecido o estado de testes a respeito de atualizações de segurança. (Debian bug 233955)
Adicionadas mais informações a respeito de conteúdo esperado em securetty (pois é específicas de kernel).
Adicionadas referências ao snoopylogger (bug da Debian 179409)
Adicionadas referências ao guarddog (bug da Debian 170710)
apt-ftparchive is in apt-utils, not in apt (thanks to Emmanuel Chantreau for pointing this out).
Removido o jvirus da lista AV.

Revisão 2-98 — JavierFernández-Sanguino Peña<jfs@debian.org>

Corrigidas URLs como sugerido por Frank Lichtenheld.
Corrigido o erro PermitRootLogin como sugerido por Stefan Lindenau.

Revisão 2-97 — September 2003 — JavierFernández-Sanguino Peña<jfs@debian.org>

Adicionadas as pessoas que fizeram as contribuições mais significantes a este manual (por favor, envie um e-mail se achar que deveria estar nesta lista e não está).
Adicionadas algumas notas a respeito de FIXME/TODOs
Movidas informações a respeito de atualizações de segurança para o inicio da seção como sugerido por Elliott Mitchell.
Adicionado o grsecurity a lista de patches do kernel para segurança, mas adicionada uma nota de rodapé sobre situações atuais como sugerido por Elliott Mitchell.
Removidos os loops (echo para 'todos') no script de segurança de rede do kernel, como sugerido por Elliott Mitchell.
Adicionadas informações (atualizadas) na seção sobre antivirus.
Reescrita da seção de proteção contra buffer overflows e adicionadas mais informações sobre patches no compilador para ativar este tipo de proteção.

Revisão 2-96 — August 2003 — JavierFernández-Sanguino Peña<jfs@debian.org>

Removido (e então novamente adicionado) apêndice sobre como rodar o Apache em ambiente chroot. O apêndice agora tem dupla licença.

Revisão 2-95 — June 2003 — JavierFernández-Sanguino Peña<jfs@debian.org>

Corrigido erros enviados por Leonard Norrgard.
Adicionada uma seção sobre como contactar o CERT para manipulação de incidentes (#after-compromise)
Mais informações sobre como tornar um proxy mais seguro.
Adicionada uma referência e removido um FIXME. Agradecimentos a Helge H. F.
Corrigido um erro (save_inactive) observado por Philippe Faes.
Corrigido diversos erros descobertos por Jaime Robles.

Revisão 2-94 — April 2003 — JavierFernández-Sanguino Peña<jfs@debian.org>

Segundo as sugestões de Maciej Stachura's, expandi a seção sobre limitação de usuários.
Corrigidos erros relatados por Wolfgang Nolte.
Corrigidos os links com o patch contribuído por Ruben Leote Mendes.
Adicionado um link para o excelente documento de David Wheeler na footnote sobre a contagem de vulnerabilidade de segurança.

Revisão 2-93 — March 2003 — FrédéricSchütz<schutz@mathgen.ch>

reescrita toda a seção sobre atributos ext2 (lsattr/chattr)

Revisão 2-92 February 2003 JavierFernández-Sanguino Peña<jfs@debian.org>, FrédéricSchütz<schutz@mathgen.ch>

União da seção 9.3 ("patches úteis do kernel") na seção 4.13 ("Adicionando patches no kernel"), e adicionado algum conteúdo.
Adicionados alguns TODOs adicionais
Adicionadas informações sobre como checar manualmente por atualizações e também sobre o cron-apt. Desta forma, o Tiger não é declarado como o único método de verificação de atualizações.
Regravação da seção sobre a execução de atualizações de segurança devido aos comentários de Jean-Marc Ranger.
Adicionada uma nota sobre a instalação da Debian (que sugere que o usuário execute uma atualização de segurança após a instalação).

Revisão 2-91 January/February 2003 JavierFernández-Sanguino Peña<jfs@debian.org>

Adicionado um patch contribuído por Frédéric Schütz.
Adicionadas algumas referências sobre capacidades. Agradecimentos a Frédéric.
Pequenas alterações na seção sobre o bind adicionando uma referência sobre a documentação on-line do BIND9 e referências apropriadas na primeira área (oi Pedro!)
Corrigida a data do changelog - ano novo :-)
Adicionada uma referência sobre o artigo do Colins para os TODOs.
Removida a referência para o antigo patch ssh+chroot
Mais patches de Carlo Perassi.
Correção de enganos (recursivo no Bind é recursão), apontados por Maik Holtkamp.

Revisão 2-9 December 2002 JavierFernández-Sanguino Peña<jfs@debian.org>

Reorganizadas informações sobre o chroot (unidas duas seções, não tem muito sentido tê-las em separado)
Adicionada as notas sober como executar o Apache em ambiente chroot por Alexandre Ratti.
Aplicação de patches contribuídos por Guillermo Jover.

Revisão 2-8 JavierFernández-Sanguino Peña<jfs@debian.org>

Aplicados patches de Carlo Perassi, as correções incluem: nova quebra de linhas, correções de URL, e corrigidos alguns FIXMEs.
Atualizado o conteúdo da FAQ do time de segurança da Debian.
Adicionado um link para a FAQ do time de segurança da Debian e da referência do desenvolvedor da Debian, as seções duplicadas podem (apenas podem) serem removidas no futuro.
Corrigida a seção sobre auditoria manual com comentários de Michal Zielinski.
Adicionado links para lista de palavras (contribuídos por Carlo Perassi)
Corrigidos alguns enganos (outros mais estão por vir).
Corrigidos links TCP como sugerido por John Summerfield.

Revisão 2-7 JavierFernández-Sanguino Peña<jfs@debian.org>

Algumas correções de erros contribuídas por Tuyen Dinh, Bartek Golenko e Daniel K. Gebhart.
Nota sobre rootkits relacionados com /dev/kmem contribuído por Laurent Bonnaud
Corrigidos enganos e FIXMEs contribuídos por Carlo Perassi.

Revisão 2-6 September 2002 CrisTillman<tillman@voicetrak.com>

Alterações para melhorar a gramática/ortografia.
s/host.deny/hosts.deny/ (1 local)
Aplicado o patch de Larry Holish's (um pouco grande, corrige diversos FIXMEs)

Revisão 2-5.1 September 2002 JavierFernández-Sanguino Peña<jfs@debian.org>

Corrigidos alguns pequenos erros enviados por Thiemo Nagel.
Adicionada uma footnote sugerida por Thiemo Nagel.
Corrigido um link de URL.

Revisão 2-5.0	August 2002	JavierFernández-Sanguino Peña<jfs@debian.org>

Aplicado um patch enviado por Philipe Gaspar com relação ao Squid que também fecha um FIXME.
Sim, outro item da FAQ com relação a banners de serviços pego da lista de segurança debian-security (thread "Telnet information" iniciada em 26 de Julho de 2002).
Adicionada uma note com relação a referências cruzadas do CVE no item da FAQ *Quanto tempo o time de segurança da Debian....*
Adicionada uma nova seção relacionada a ataques ARP contribuída por Arnaud "Arhuman" Assad.
Novo item da FAQ com relação ao dmesg e logind e console pelo kernel.
Pequenos detalhes de informações relacionadas a checagem de assinaturas em pacotes (ele parecia não ter uma versão beta passada).
Novo item da FAQ com relação a falso positivo de ferramentas de checagem de vulnerabilidades.
Adicionadas nova seções ao capítulo que contém informações sobre assinatura de pacotes e reorganizando-as como um novo capítulo sobre *Infraestrutura de Segurança na Debian*.
Novo ítem da FAQ com uma comparação da Debian com outras distribuições Linux.
Nova seção sobre agentes de mensagem de usuários com funcionalidade GPG/PGP no capítulo ferramentas de segurança.
Esclarecimentos de como ativa senhas MD5 na woody, adicionadas referências à PAM assim também como uma nota relacionada a definição de max na PAM.
Adicionado um novo apêndice sobre como criar um ambiente chroot (após brigar um pouco com makejail e corrigindo, também, alguns de seus bugs), integradas informações duplicadas em todo o apêndice.
Adicionadas mais informações relacionadas ao chroot de **SSH** e seu impacto na transferência segura de arquivos. Algumas informações que foram pegas da lista de discussão debian-security (da thread de Junho de 2002: *transferências de arquivos seguras*).
Novas seções sobre como fazer atualizações automáticas em sistemas Debian assim também como dicas de uso de testing ou unstable relacionadas com atualizações de segurança.
Nova seção relacionada sobre como manter-se atualizado com patches de segurança na seção *Antes do comprometimento* assim como uma nova seção sobre a lista de discussão debian-security-announce mailing.
Adicionadas informações sobre como gerar automaticamente senhas fortes.
Nova seção com relação a usuários inativos.
Reorganização da seção sobre como tornar um servidor de mensagens mais seguro com base na discussão sobre instalação *Segura/fortalecida/mínima da Debian (Ou "Porque o sistema básico é do jeito que é?")* que ocorreu na lista debian-security (em Maio de 2002).
Reorganização da seção sobre parâmetros de rede do kernel, com dados fornecidos pela lista de discussão debian-security (em Maio de 2002, *syn flood attacked?*) e também adicionado um novo ítem da FAQ.
Nova seção sobre com verificar senhas de usuarios e que pacotes instalar para fazer isto.
Nova seção sobre a criptografia PPTP com clientes Microsoft discutido na lista debian-security (em Abril de 2002).
Adicionada uma nova seção descrevendo que problemas existem quando direciona um serviço a um endereço IP específico, esta informação foi escrita baseada em uma lista de discussão da bugtraq com a thread: *Linux kernel 2.4 "weak end host" (anteriormente discutida na debian-security como "problema no")* (iniciada em 9 de Maio de 2002 por Felix von Leitner).
Adicionadas informações sobre o protocolo versão 2 do **ssh**.
Adicionadas duas sub-seções relacionadas a configurações seguras do Apache (coisas específicas a Debian, é claro).
Adicionada uma nova FAQ relacionada a soquetes simples, um relacionado a /root, um ítem relacionado ao grupo users e outra relacionada a log e permissões de arquivos de configuração.
Adicionada uma referência ao problem na libpam-cracklib que ainda pode estar aberto... (precisa ser verificado)
Adicionadas mais informações com relação a análise forense (pendente mais informações sobre ferramentas de inspeção de pacotes como **tcpflow**).
Alterado o ítem "o que posso fazer com relação a comprometimento" na listagem e adicionado mais conteúdo.

Adicionadas mais informações sobre como configurar o Xscreensaver para bloquear a tela automaticamente após um tempo limite estabelecido.
Adicionada uma nota relacionada a utilitários que não deve instalar no sistema. Inclui uma nota relacionada ao Perl e porque ele não pode ser facilmente removido da Debian. A idéia veio após ler documentos do Intersects relacionado com o fortalecimento do Linux.
Adicionadas informações sobre o lvm e sistemas de arquivos com journaling, o ext3 é recomendado. No entanto, as informações lá devem ser muito genérica.
Adicionado um link para a versão texto on-line (verificar).
Adicionados mais alguns materiais com relação a informações sobre como fazer um firewall em um sistema local, levado por um comentário feito por Huber Chan na lista de discussão.
Adicionadas mais informações sobre a limitação do PAM e ponteiros aos documentos de Kurt Seifried's (relacionada a uma postagem por ele na bugtrack em 4 de Abril de 2002 respondendo a uma pessoa que "descobriu" uma vulnerabilidade na Debian GNU/Linux relacionada a esgotamento de recursos).
Como sugerido por Julián Muñoz, fornecidas mais informações sobre a umask padrão da Debian e o que um usuário pode acessar se tiver um shell em um sistema (provided more information on the default Debian umask and what a user can access if he has been given a shell in the system
Incluir uma nota na seção sobre senha de BIOS devido a um comentário de Andreas Wohlfeld.
Incluir patches fornecido por Alfred E. Heggestad corrigindo muitos dos erros ainda presentes no documento.
Adicionada uma referência ao changelog na seção créditos, pois muitas pessoas que contribuem estão listadas aqui (e não lá).
Adicionadas algumas notas a mais sobre a seção chattr e uma nova seção após a instalação falando sobre snapshots do sistema. Ambas idéias foram contribuídas por Kurt Pomeroy.
Adicionada uma nova seção após a instalação apenas para lembrar os usuários de alterar a seqüência de partida.
Adicionados alguns itens a mais no TODO, fornecidos por Korn Andras.
Adicionado uma referência as regras do NIST sobre como tornar o DNS mais seguro, fornecidas por Daniel Quinlan.
Adicionado um pequeno parágrafo relacionado com a infraestrutura de certificados SSL da Debian.
Adicionadas sugestões de Daniel Quinlan's com relação a autenticação **ssh** e configuração de relay do exim.
Adicionadas mais informações sobre como tornar o bind mais seguro incluindo alterações propostas por Daniel Quinlan e um apêndice com um script para fazer algumas das alterações comentadas naquela seção.
Adicionado um ponteiro a outro ítem relacionado a fazer chroot do Bind (precisam ser unidas).
Adicionada uma linha contribuída por Cristian Ionescu-Idbohrn para pegar pacotes com o suporte a tcpwrappers.
Adicionada um pouco mais de informações sobre a configuração padrão de PAM da Debian.
Incluída uma questão da FAQ sobre o uso de PAM para fornecer serviços sem contas shell.
movidos dois itens da FAQ para outra seção e adicionada uma nova FAQ relacionada com detecção de ataques (e sistemas comprometidos).
Incluídas informações sobre como configurar uma firewall ponte (incluindo um Apêndice modelo). Obrigado a Francois Bayart quem enviou isto para mim em Março.
Adicionada uma FAQ relacionada com o syslogd *MARK heartbeat* de uma questão respondida por Noah Meyerhans e Alain Tesio em Dezembro de 2001.
Incluídas informações sobre proteção contra buffer overflow assim como mais informações sobre patches de kernel.
Adicionadas mais informações e reorganização da seção sobre firewall. Atualização da informação com relação ao pacote iptables e geradores de firewall disponíveis.
Reorganização das informações disponíveis sobre checagem de logs, movidas as informações sobre checagem de logs de detecção de intrusão de máquinas para aquela seção.
Adicionadas mais informações sobre como preparar um pacote estático para o bind em chroot (não testado).
Adicionado um ítem da FAQ relacionado com servidores/serviços mais específicos (podem ser expandidos com algumas das recomendações da lista debian-security).
Adicionadas mais informações sobre os serviços RPC (e quando são necessários).

Adicionadas mais informações sobre capacidades (e o que o lcap faz). Existe alguma boa documentação sobre isto? e não encontrei qualquer documentação em meu kernel 2.4
Corrigidos alguns enganos.

Revisão 2-4 June 2002 JavierFernández-Sanguino Peña<jfs@debian.org>

Parte da seção sobre BIOS foi reescrita

Revisão 2-3.1 April 2002 JavierFernández-Sanguino Peña<jfs@debian.org>

Trocadas algumas localizações de arquivos com a tage de arquivo.
Corrigido problema notificado por Edi Stojicevi.
Leve alteração na seção sobre ferramentas de auditoria remota.
Adicionados alguns itens para fazer.
Adicionadas mais informações com relação a impressoras e o arquivo de configuração do cups (pego de uma thread na debian-security).
Adicionado um patch enviado por Jesus Climent com relação ao acesso de usuários válidos ao sistema no proftpd quando se está configurando um servidor anônimo.
Pequena alteração nos esquemas de partição para o caso especial de servidores de mensagens.
Adicionado uma referência do livro "Hacking Linux Exposed" na seção livros.
Corrigido um erro de diretório notificado por Eduardo Pérez Ureta.
Corrigido um erro na checklist do /etc/ssh observado por Edi Stojicevi.

Revisão 2-3.0 April 2002 JavierFernández-Sanguino Peña<jfs@debian.org>

Corrigida a localização do arquivo de configuração do dpkg.
Remoção do Alexander das informações de contato.
Adicionado um endereço alternativo de e-mails.
Corrigido o endereço de e-mail do Alexander
Corrigida a localização das chaves de lançamento (agradecimentos a Pedro Zorzenon por nos apontar isto).

Revisão 2-2 April 2002 JavierFernández-Sanguino Peña<jfs@debian.org>

Corrigidos problemas, agradecimentos a Jamin W. Collins.
Adicionada uma referência a página de manual do apt-extracttemplate (documenta a configuração do APT::ExtractTemplate).
Adicionada uma seção sobre o SSH restrito. Informações baseadas naquilo postadas por Mark Janssen, Christian G. Warden e Emmanuel Lacour na lista de segurança debian-security.
Adicionadas informações sobre programas de anti-vírus.
Adicionada uma FAQ: logs do su devido a execução do cron como usuário root.

Revisão 2-1 April 2002 JavierFernández-Sanguino Peña<jfs@debian.org>

Alterado o FIXME a partir do lshell agradecimentos a Oohara Yuuma.
Adicionados pacote a sXid e removido comentário pois ele *está* disponível.
Corrigido um número de erros descobertos por Oohara Yuuma.
ACID está agora disponível na Debian (a partir do pacote acidlab) obrigado a Oohara Yuuma por notificar isto.
Correção dos links da LinuxSecurity (agradecimentos a Dave Wreski pelo aviso).

Revisão 2-0 March 2002 JavierFernández-Sanguino Peña<jfs@debian.org>

Conversão do HOWTO em um Manual (agora eu poderei propriamente dizer RTFM)
Adicionadas mais informações com relação ao tcp wrappers e a Debian (agora mutos serviços são compilados com suporte a eles assim não será mais um assunto relacionado ao **inetd**).
Esclarecidas informações sobre a desativação de serviços para torna-lo mais consistente (informações sobre o rpc ainda referenciadas ao update-rc.d)
Adicionada uma pequena nota sobre o lprng.
Adicionadas ainda mais informações sobre servidores comprometidos (ainda de uma forma grossa),
Corrigidos problemas reportados por Mark Bucciarelli.

Adicionados mais alguns passos sobre a recuperação de senhas para cobrir os casos onde o administrador tem a opção "paranoid-mode=on" ativada.
Adicionadas mais informações sobre como definir "paranoid-mode=on" quando executa o logon em um console.
Novo parágrafo para introduzir configuração de serviços.
Reorganização da seção *Após a instalação* assim ela será quebradas em diversos assuntos e será facilmente lida.
Escrever informações sobre como configurar firewalls com a configuração padrão da Debian 3.0 (pacote iptables).
Pequeno parágrafo explicando porque a instalação conectada a Internet não é uma boa idéia e como evitar isto usando ferramentas da Debian.
Pequeno parágrafo sobre patching referenciando um paper do IEEE.
Apêndice de como configurar uma máquina Debian com o snort, baseada no que Vladimir enviou para a lista de segurança debian-security (em 3 de Setembro de 2001)
Informações sobre como o logcheck é configurado na Debian e como ele pode ser configurado para realizar HIDS.
Informações sobre contabilização de usuários e análise de perfis.
Incluída a configuração do apt.conf para /usr somente leitura copiada da postagem de Olaf Meeuwissen's para a lista de discussão debian-security.
Nova seção sobre VPN com alguns ponteiros e pacotes disponíveis na Debian (precisa conteúdo sobre como configurar VPNs e assuntos específicos da Debian), baseada na postagem de Jaroslaw Tabor's e Samuli Suonpaa's post na lista debian-security.
Pequena nota com relação a alguns programas que podem construir automaticamente jaulas chroot.
Novo ítem da FAQ relacionado a ident, baseado em uma discussão na lista de discussão debian-security (em Fevereiro de 2002, iniciada por Johannes Weiss).
Novo ítem da FAQ com relação ao **inetd** baseada em uma discussão da lista de discussão debian-security (em Fevereiro de 2002).
Introduzida uma nota ao rcconf na seção "desabilitando serviços".
Variação da abordagem com relação a LKM, agradecimentos a Philipe Gaspar
Adicionado ponteiros a documentos do CERT e recursos

Revisão 1-99 January 2002 JavierFernández-Sanguino Peña<jfs@debian.org>

Adicionado um novo ítem da FAQ com relação ao tempo para se corrigir vulnerabilidades de segurança.
Reorganizada as seções da FAQ.
Iniciada a escrita de uma seção com relação a firewall na Debian GNU/Linux (pode ser um pouco alterada).
Corrigidos problemas enviados por Matt Kraai
Corrigidas informações relacionadas a DNS
Adicionadas informações sobre o whisker e nbtscan na seção auditoria.
Corrigidas algumas URLs incorretas

Revisão 1-98 January 2002 JavierFernández-Sanguino Peña<jfs@debian.org>

Adicionada uma nova seção sobre auditoria usando Debian GNU/Linux.
Adicionadas informações relacionadas ao daemon do finger a partir da lista de segurança.

Revisão 1-97 January 2002 JavierFernández-Sanguino Peña<jfs@debian.org>

Corrigido o link para o Linux trustees
Corrigidos problemas (patches de Oohara Yuuma e Pedro Zorzenon)

Revisão 1-96 December 2001 JavierFernández-Sanguino Peña<jfs@debian.org>

Reorganizada a instalação e remoção de serviços e adicionadas algumas novas notas.
Adicionadas algumas notas com relação ao uso de verificadores de integridade como ferramentas de detecção de intrusão.
Adicionado um capítulo com relação a assinatura de pacotes.

Revisão 1-95 December 2001 JavierFernández-Sanguino Peña<jfs@debian.org>
Adicionadas notas com relação a segurança no Squid enviada por Philipe Gaspar.
Corrigido os links sobre rootkit agradecimentos a Philipe Gaspar.
Revisão 1-94 November 2001 JavierFernández-Sanguino Peña<jfs@debian.org>
Adicionadas algumas notas com relação ao Apache e o Lpr/lprng.
Adicionadas mais informações com relação as partições noexec e read-only.
Reescrita a parte sobre como os usuários podem ajudar a Debian em assuntos relacionados a segurança (item da FAQ).
Revisão 1-93 November 2001 JavierFernández-Sanguino Peña<jfs@debian.org>
Corrigida a localização do programa mail.
Adicionados alguns novos itens na FAQ.
Revisão 1-92 October 2001 JavierFernández-Sanguino Peña<jfs@debian.org>
Adicionada uma pequena seção sobre como a Debian trabalha com a segurança
Esclarecimentos sobre as senhas MD5 (agradecimentos a "rocky")
Adicionadas mais informações com relação ao harden-X de Stephen van Egmond
Adicionados alguns novos itens na FAQ.
Revisão 1-91 October 2001 JavierFernández-Sanguino Peña<jfs@debian.org>
Adicionadas mais informações de forense enviadas por Yotam Rubin.
Adicionadas informações sobre como construir um honeypot usando a Debian GNU/Linux.
Adicionados alguns TODOS a mais.
Corrigidos mais problemas (agradecimentos a Yotam!)
Revisão 1-9 October 2001 JavierFernández-Sanguino Peña<jfs@debian.org>
Adicionado um patch para corrigir problemas de escrita e algumas informações novas (contribuídas por Yotam Rubin)
Adicionadas referências a outras documentações online (e offline) ambas na seção (veja “Conhecimento necessário”) e junto com o texto em outra seções.
Adicionadas algumas informações sobre a configuração de opções do Bind para restringir o acesso ao servidor DNS.
Adicionadas informações sobre como fortalecer automaticamente um sistema Debian (com relação ao pacote harden e o bastille).
Removido alguns TODOs fechados e adicionados alguns novos.
Revisão 1-8 October 2001 JavierFernández-Sanguino Peña<jfs@debian.org>
Adicionada a lista padrão de usuários/grupos fornecidas por Joey Hess a lista de discussão debian-security.
Adicionadas informações a respeito de root-kits LKM (“Loadable Kernel Modules (LKM)”) contribuído por Philipe Gaspar.
Adicionada informação a respeito do Proftp contribuído por Emmanuel Lacour.
Apêndice de checklist recuperado de Era Eriksson.
Adicionados alguns novos itens na lista TODO e removidos outros.
Incluir manualmente os patches e Era pois nem todos foram incluídos na seção anterior.
Revisão 1-7 September 2001 JavierFernández-Sanguino Peña<jfs@debian.org>, EraEriksson<era@iki.fi>
Correção de erros e alterações de palavras
Pequenas mudanças em tags para manter a remoção de tags tt e sua substituição por tags prgn/package.
Revisão 1-6 August 2001 JavierFernández-Sanguino Peña<jfs@debian.org>

Adicionado ponteiro para documentos como publicado na DDP (deverá substituir o original em um futuro próximo).
Iniciada uma mini-FAQ (deverá ser expandida) com algumas questões recuperadas de minha caixa de mensagens.
Adicionadas informações gerais que devem ser consideradas durante a segurança.
Adicionado um parágrafo relacionado a entrega de mensagens locais (entrada).
Adicionadas algumas referências a mais informações.
Adicionadas informações com relação ao serviço de impressão.
Adicionada uma lista de checagem de fortalecimento de segurança.
Reorganizadas as informações a respeito de NIS e RPC.
Adicionadas mais novas notas durante a leitura deste documento em meu novo visor :-)
Corrigidas algumas linhas mal formatadas.
Corrigidos alguns enganos.
Adicionada a idéia do Genus/Paranóia contribuída por Gaby Schilders.

Revisão 1-5 — May 2001 — JavierFernández-Sanguino Peña<jfs@debian.org>, JosipRodin<joy@debian.org>

Adicionados parágrafos relacionados ao BIND e alguns FIXMEs.

Revisão 1-4 — May 2001 — JavierFernández-Sanguino Peña<jfs@debian.org>

Pequeno parágrafo sobre checagem de setuid
Várias pequenas limpezas
Encontrado como usar o `sgml2txt -f` para fazer a versão texto

Revisão 1-3 — March 2001 — JavierFernández-Sanguino Peña<jfs@debian.org>

Adicionada uma atualização de segurança após o parágrafo de instalação
Adicionado um parágrafo relacionado ao proftpd
Agora realmente escrevia algo sobre o XDM, desculpe pelo atraso

Revisão 1-2 — December 2000 — JavierFernández-Sanguino Peña<jfs@debian.org>

Várias correções gramaticais feitas por James Treacy, novo parágrafo sobre o XDM

Revisão 1-1 — December 2000 — JavierFernández-Sanguino Peña<jfs@debian.org>

Correção de erros, adições diversas

Revisão 1-0 — December 2000 — JavierFernández-Sanguino Peña<jfs@debian.org>

Lançamento inicial

Apêndice B. Appendix

Passo-a-passo do processo de fortalecimento

Abaixo está uma pós-instalação, um procedimento passo-a-passo para tornar no sistema Debian 2.2 GNU/Linux mais seguro. Esse procedimento é uma alternativa para tornar os serviços de redes mais seguros. Será mostrado o processo completo do que deve ser feito durante a configuração. Também, veja "Checklist de configuração".

- Instale o sistema, levando em conta as informações sobre o particionamento que foi citada anteriormente neste documento. Depois da instalação básica, vá à instalação personalizada. Não selecione os pacotes de tarefa. Selecione senhas no formato shadow.

- Usando **dselect**, exclua todos os pacotes desnecessários, exceto os selecionados, antes de proceder com o [I]nstall. Mantenha um número reduzido de pacotes para o sistema.

- Atualize todos os softwares para a última versão disponível dos pacotes em security.debian.org como explicado anteriormente em "Executar uma atualização de segurança".

- Implementar as sugestões apresentadas neste manual com relação às cotas de usuários, definições de login e **lilo**

- Fazer uma lista de serviços que estão rodando no seu sistema. Tente:

  ```
  $ ps -aux
  $ netstat -pn -l -A inet
  # /usr/sbin/lsof -i | grep LISTEN
  ```

  Você precisará instalar o lsof-2.2 para o terceiro comando acima funcionar (execute como super-usuário). Você deve estar ciente de que o **lsof** pode traduzir a palavra LISTEN para suas configurações de localização.

- Para excluir serviços desnecessários, primeiro determine qual pacote fornece o serviço e como ele é inicializado. Isto pode ser feito verificando os programas que escutam no soquete. O shell script abaixo, que utiliza os programas **lsof** e **dpkg**, faz isso:

  ```
  #!/bin/sh
  # FIXME: this is quick and dirty; replace with a more robust script snippet
  for i in `sudo lsof -i | grep LISTEN | cut -d " " -f 1 |sort -u` ; do
\t  pack=`dpkg -S $i |grep bin |cut -f 1 -d : | uniq`
\t  echo "Service $i is installed by $pack";
\t  init=`dpkg -L $pack |grep init.d/ `
\t  if [ ! -z "$init" ]; then
\t\t   echo "and is run by $init"
\t  fi
  done
  ```

- Se você encontrar algum serviço desnecessário, exclua o pacote associado (com **dpkg --purge**), ou desabilite a inicialização automática durante a fase de boot usando o comando **update-rc.d** (veja "Desabilitando daemons de serviço").

- Para os serviços inetd (iniciados pelo superdaemon), verifique quais serviços estão ativados em `/etc/inetd.conf` através de:

```
$ grep -v "^#" /etc/inetd.conf | sort -u
```

  Então desative estes serviços desnecessários comentando a linha referente em `/etc/inetd.conf`, excluindo o pacote ou utilizando o comando **update-inetd**.

- Se você utiliza serviços wrapped (aqueles que utilizam **/usr/sbin/tcpd**), verifique se os arquivos `/etc/hosts.allow` e `/etc/hosts.deny` são configurados de acordo com sua política de serviço.

- Se o servidor usa mais que uma interface externa, dependendo do seu serviço, você pode limitar o serviço para escutar em uma interface específica. Por exemplo, se você quiser somente acesso interno para o FTP, você deve configurar o daemon FTP para escutar somente na sua interface de gerência, não em todas interfaces (i.e, 0.0.0.0:21).

- Reinicie o computador, ou troque o modo de single user para multiuser usando os comandos:

```
$ init 1
(....)
$ init 2
```

- Então verifique agora os serviços que estão disponíveis, e se necessário, repita os passos acima.

- Agora instale os serviços necessários, se não tiver feito isso ainda, e os configure corretamente.

- Use o comando shell abaixo para determinar com que usuário cada serviço disponível está sendo executado:

```
$ for i in `/usr/sbin/lsof -i |grep LISTEN |cut -d " " -f 1 |sort -u`; \
> do user=`ps -ef |grep $i |grep -v grep |cut -f 1 -d " "` ; \
> echo "Service $i is running as user $user"; done
```

  Considere alterar esses serviços para um usuário/grupo específico e talvez até enjaulá-los (**chroot**'ing) para aumentar nível de segurança. Você pode fazer isto alterando os scripts de inicialização em `/etc/init.d`. A maioria dos serviços no Debian usa o **start-stop-daemon** com as opções (`--change-uid` e `--chroot`) para fazer isso. Uma observação com relação ao enjaulamento (**chroot**'ing) dos serviços: você precisa colocar todos os arquivos instalados pelo pacote (use dpkg -L) que fornece o serviço, assim como qualquer pacote dependente, na jaula **chroot**. Informações sobre a configuração de um ambiente **chroot** para o programa **ssh** podem ser encontrada em “Chroot environment for SSH”.

- Repita os passos acima para certificar que somente os serviços desejados estejam rodando e esteja sendo usada a combinação de usuário/grupo correta.

- Teste os serviços instalados para ver se estão funcionando corretamente.

- Verifique o sistema usando um vulnerability assessment scanner (tipo o nessus), para determinar as vulnerabilidades no sistema (i.e., mal-configuração, serviços antigos e desnecessários).

- Instale ferramentas de detecção de intrusão de rede e host como snort e logsentry.

- Repita o passo de varredura da rede e verifique se os sistemas de detecção de intrusão estão funcionando corretamente.

Para paranóia real, também considere o seguinte:

- Adicione as capacidades de firewall do sistema, conexões de entrada só devem ser feitas para os serviços oferecidos e limite as conexões de saída somente para aqueles que são autorizados.
- Verifique novamente a instalação com uma nova vulnerability assessment usando um varredor de rede.
- Usando um varredor de rede, verifique as conexões de saídas do sistema para um host remoto e certifique-se que as conexões indesejadas sejam estabelecida.

FIXME: este procedimento engloba o fortalecimento de serviços, mas não o fortalecimento a nível de usuário, incluindo informações sobre verificação de permissões de usuários, arquivos SETUID e congelamento de alterações no sistema utilizando o sistema de arquivo ext2.

Checklist de configuração

Este apêndice retrata resumidamente os pontos de outras seções neste manual em um checklist no formato. A idéia é disponibilizar um sumário para a pessoa que já leu o manual buscar uma informação rapidamente. Existem outros checklists bons disponíveis, incluindo o http://seifried.org/security/os/linux/20020324-securing-linux-step-by-step.html de Kurt Seifried e http://www.cert.org/tech_tips/usc20_full.html.

FIXME: Isso é baseado na versão 1.4 do manual e talvez precise de atualização.

- Limite o acesso físico e as capacidade de inicialização
  - Enable a password in the BIOS.
  - Disable floppy/cdrom/... booting in the system's BIOS.
  - Configure uma senha para o LILO ou GRUB (`/etc/lilo.conf` ou `/boot/grub/menu.lst`, respectivamente); verifique se o arquivo de configuração do LILO ou GRUB está protegido contra gravação.
- Particionamento
  - Separe os dados de escrita do usuário, dados que não são do sistema, e dados que são trocados rapidamente em tempo de execução para suas próprias partições
  - Set `nosuid,noexec,nodev` mount options in `/etc/fstab` on ext2/3 partitions that should not hold binaries such as `/home` or `/tmp`.
- Higiene de senhas e segurança no login
  - Configure uma senha segura para o super-usuário
  - Instale e use o PAM
    - Adicione suporte MD5 para o PAM e tenha certeza que (falando de forma generalizada) as entradas nos arquivos em `/etc/pam.d/` que garantem acesso à máquina tenham o segundo campo configurado como `requisite` ou `required`.
    - Modifique o `/etc/pam.d/login` para permite somente logins locais para o super-usuário.
    - Também marque tty:s autorizado em `/etc/security/access.conf` e geralmente configure este arquivo para limitar ao máximo possível o login do super-usuário.

    - Adicione o módulo pam_limits.so se você deseja configurar os limites por usuários
    - Modifique `/etc/pam.d/passwd`: configure o tamanho mínimo para as senhas (6 caracteres talvez) e ative o MD5
    - Adicione o grupo wheel para `/etc/group` se desejar; adicione a entrada pam_wheel.so group=wheel para `/etc/pam.d/su`
    - Para controles customizados por usuários, utilize o módulo pam_listfile.so
    - Tenha um arquivo `/etc/pam.d/other` e o configure com um grau de segurança reforçado
  - Configure limites em `/etc/security/limits.conf` (note que `/etc/limits` não é usado se você já estiver usando o PAM)
  - Aumente a segurança em `/etc/login.defs`; também, se você ativar o MD5 e/ou PAM, tenha certeza de fazer também as alterações correspondentes aqui, também
  - Tighten up `/etc/pam.d/login`
  - Desative o acesso ftp ao super-usuário em `/etc/ftpusers`
  - Disable network root login; use su(1) or sudo(1). (consider installing sudo)
  - Usar o PAM para reforçar barreiras adicionais aos logins?
- Outras questões de segurança local
  - Modificações no kernel (veja “Configurando características de rede do kernel”)
  - Patches no Kernel (veja “Adicionando patches no kernel”)
  - Tighten up log file permissions (`/var/log/{last,fail}log`, Apache logs)
  - Certifique-se que a verificação SETUID está ativada em `/etc/checksecurity.conf`
  - Considere configurar alguns arquivos de logs como somente append e os arquivo de configuração imutáveis, usando o comando chattr (somente para arquivos ext2)
  - Configurar a integridade de arquivo (veja “Verificando a integridade do sistema de arquivos”). Instale debsums
  - Efetuar o log de tudo em uma impressora local?
  - Gravar suas configurações em um CD inicializável e boof off?
  - Desativar os módulos do kernel?
- Limitar acesso a rede
  - Instale e configure **ssh** (sugiro PermitRootLogin No em `/etc/ssh/sshd_config`, PermitEmptyPasswords No; note outras sugestões também no texto)
  - Disable or remove **in.telnetd**, if installed
  - Geralmente, desative serviços desnecessários em `/etc/inetd.conf` usando o comando **update-inetd --disable** (ou desative **inetd** completamente, ou use o um substituto como **xinetd** ou **rlinetd**)

    - Disable other gratuitous network services; ftp, DNS, WWW etc should not be running if you do not need them and monitor them regularly. In most cases mail should be running but configured for local delivery only.

    - Para aqueles serviços que você precisa, não use os programas mais comuns, procure por versões mais seguras distribuídas com o Debian (ou de outras fontes). Seja lá o que você for parar de executar, tenha certeza que você entende os riscos.

    - Configure jaula **chroot** para usuários externos e daemons.

    - Configure firewall and tcpwrappers (i.e. hosts_access(5)); note trick for `/etc/hosts.deny` in text.

    - Se você executa o ftp, configure seu servidor ftpd sempre para executar enjaulado para o diretório home dos usuários

    - Se você executa X, desative a autenticação xhost e use-o com **ssh**; melhor ainda, se puder desative o X (adicione -nolisten tcp para a linha de comando do X e desligue o XDMCP no `/etc/X11/xdm/xdm-config` configurando requestPort para 0)

    - Desative acesso externo para as impressoras

    - Use tunelamento para qualquer sessão IMAP ou POP através do SSL ou **ssh**; instale stuneel se você quer fornecer este serviços para usuários de mail externos

    - Configure um host de log e configure as outras máquinas para enviar logs para esse host (`/etc/syslog.conf`)

    - Torne seguro o BIND, Sendmail, e outros daemons complexos (execute-os com uma jaula**chroot**; execute como um pseudo-usuário não root)

    - Install tiger or a similar network intrusion detection tool.

    - Install snort or a similar network intrusion detection tool.v

    - Faça sem NIS ou RPC se puder (desative portmap).

- Políticas de segurança

    - Eduque os usuários sobre os porquês e comos de suas políticas. Quando você proíbe algo que está disponível regularmente em outros sistemas, forneça uma documentação que explique como obter resultados similares através de outros meios mais seguros.

    - Proíba o uso de protocolos que utilizam senhas em texto plano (**telnet**, **rsh** e similares; ftp, imap, http, ...).

    - Proíba programas que usam SVGAlib.

    - Use cotas de disco.

- Mantenha-se informado sobre questões relacionadas à segurança

    - Inscreva-se em listas de discussão sobre segurança

    - Configure apt para atualização de segurança -- adicione no arquivo `/etc/apt/sources.list` uma entrada (ou entradas) para http://security.debian.org/debian-security

- Também lembre-se de executar periodicamente os comandos **apt-get update ; apt-get upgrade** (talvez instalar como um job no **cron**?) como explicado em “Executar uma atualização de segurança”.

Configurando um IDS stand-alone

You can easily set up a dedicated Debian system as a stand-alone Intrusion Detection System using snort and a web-based interface to analyse the intrusion detection alerts:

- Instale um sistema Debian base e não selecione nenhum pacote adicional.
- Install one of the Snort versions with database support and configure the IDS to log alerts into the database.
- Download and install BASE (Basic Analysis and Security Engine), or ACID (Analysis Console for Intrusion Databases). Configure it to use the same database than Snort.
- Download and install the necessary packages[1].

BASE is currently packaged for Debian in acidbase and ACID is packaged as acidlab[2]. Both provide a graphical WWW interface to Snort's output.

Besides the base installation you will also need a web server (such as apache), a **PHP** interpreter and a relational database (such postgresql or mysql) where Snort will store its alerts.

Este sistema deve ser configurado com pelo menos duas interfaces de rede; uma interface conectada ao gerenciamento da LAN (para acessar os resultados e suporte do sistema), e outra interface sem nenhum endereço IP anexada ao segmento de rede a ser analisado.

You should configure both interfaces in the standard Debian `/etc/network/interfaces` configuration file. One (the management LAN) address can be configured as you would normally do. The other interface needs to be configured so that it is started up when the system boots, but with no interface address. You can use the following interface definition:

```
auto eth0
iface eth0 inet manual
      up ifconfig $IFACE 0.0.0.0 up
      up ip link set $IFACE promisc on
      down ip link set $IFACE promisc off
      down ifconfig $IFACE down
```

The above configures an interface to read all the traffic on the network in a *stealth*-type configuration. This prevents the NIDS system to be a direct target in a hostile network since the sensors have no IP address on the network. Notice, however, that there have been known bugs over time in sensors part of NIDS (for example see https://lists.debian.org/debian-security-announce/2003/msg00087.html related to Snort) and remote buffer overflows might even be triggered by network packet processing.

You might also want to read the http://www.faqs.org/docs/Linux-HOWTO/Snort-Statistics-HOWTO.html and the documentation available at the https://www.snort.org/#documents.

[1] Typically the needed packages will be installed through the dependencies
[2] It can also be downloaded from http://www.cert.org/kb/acid/, http://acidlab.sourceforge.net or http://www.andrew.cmu.edu/~rdanyliw/snort/.

Configurando uma ponte firewall

Esta informação foi contribuição de Francois Bayart para ajudar os usuário a configurar um Linux como ponte/firewall com o kernel 2.4.x e iptables. Patches do kernel não são mais necessários, uma vez que o código passou a fazer parte do kernel do Linux.

Para configurar o kernel com o suporte necessário, execute `make menuconfig` ou `make xconfig`. Na seção *Networking options*, ative as seguintes opções:

```
[*] Network packet filtering (replaces ipchains)
[ ]   Network packet filtering debugging (NEW)
<*> 802.1d Ethernet Bridging
[*]   netfilter (firewalling) support (NEW)
```

Cuidado: você deve desativar isso se você quiser aplicar algumas regras de firewall ou o **iptables** não funcionará:

```
[ ]   Network packet filtering debugging (NEW)
```

Próximo passo, adicione as opções corretas na seção *IP: Netfilter Configuration*. Então, compile e instale o kernel. Se você quiser fazer isso no *jeito do Debian*, instale o kernel-package e execute **make-kpkg** para criar um pacote Debian customizado do kernel que possa ser instalado no servidor usando o dpkg. Uma vez que o novo kernel é compilado e instalado, instale o pacote bridge-utils.

Quando estes passos forem feitos, você pode completar a configuração de sua ponte. A próxima seção apresenta duas possíveis configurações para a ponte, cada uma com um mapa de rede hipotético e os comandos necessários.

Uma ponte fornecendo capacidades de NAT e firewall

A primeira configuração usa a ponte como um firewall com tradução de endereços de rede (NAT) que protege o servidor e os clientes da rede interna. Um diagrama da configuração da rede é mostrado abaixo:

```
Internet ---- router ( 62.3.3.25 ) ---- bridge (62.3.3.26 gw 62.3.3.25 / 192.168.0
                                          |
                                          |
                                          |---- WWW Server (62.3.3.27 gw 62.3.3.25
                                          |
                                          |
                                        LAN --- Zipowz (192.168.0.2 gw 192.168.0.
```

Os seguintes comando mostram como esta ponte pode ser configurada.

```
# Create the interface br0
/usr/sbin/brctl addbr br0

# Add the Ethernet interface to use with the bridge
/usr/sbin/brctl addif br0 eth0
/usr/sbin/brctl addif br0 eth1

# Start up the Ethernet interface
```

```
/sbin/ifconfig eth0 0.0.0.0
/sbin/ifconfig eth1 0.0.0.0

# Configure the bridge ethernet
# The bridge will be correct and invisible ( transparent firewall ).
# It's hidden in a traceroute and you keep your real gateway on the
# other computers. Now if you want you can config a gateway on your
# bridge and choose it as your new gateway for the other computers.

/sbin/ifconfig br0 62.3.3.26 netmask 255.255.255.248 broadcast 62.3.3.32

# I have added this internal IP to create my NAT
ip addr add 192.168.0.1/24 dev br0
/sbin/route add default gw 62.3.3.25
```

Uma ponte fornecendo capacidades de firewall

Uma segunda possível configuração é um sistema que funciona como um firewall transparente para a LAN com um espaço de endereços IP públicos.

```
Internet ---- router (62.3.3.25) ---- bridge (62.3.3.26)
                                          |
                                          |
                                          |---- WWW Server (62.3.3.28 gw 62.3.3.25)
                                          |
                                          |
                                          |---- Mail Server (62.3.3.27 gw 62.3.3.25)
```

Os seguintes comando mostram como esta ponte pode ser configurada.

```
# Create the interface br0
/usr/sbin/brctl addbr br0

# Add the Ethernet interface to use with the bridge
/usr/sbin/brctl addif br0 eth0
/usr/sbin/brctl addif br0 eth1

# Start up the Ethernet interface
/sbin/ifconfig eth0 0.0.0.0
/sbin/ifconfig eth1 0.0.0.0

# Configure the bridge Ethernet
# The bridge will be correct and invisible ( transparent firewall ).
# It's hidden in a traceroute and you keep your real gateway on the
# other computers. Now if you want you can config a gateway on your
# bridge and choose it as your new gateway for the other computers.

/sbin/ifconfig br0 62.3.3.26 netmask 255.255.255.248 broadcast 62.3.3.32
```

Se você seguir as rotas para o Linux Mail Server, não enxergará a ponte. Se você quiser acessar a ponte com o **ssh**, você deve ter um gateway ou acessar um outro servidor, como o "Mail Server", e então conectar à ponte através de uma placa de rede interna.

Regras básicas do IPtables

As regras básicas a seguir podem ser usadas em qualquer uma das duas configurações mostradas acima.

Exemplo B.1. Regras básicas do IPtables

```
iptables -F FORWARD
iptables -P FORWARD DROP
iptables -A FORWARD -s 0.0.0.0/0.0.0.0 -d 0.0.0.0/0.0.0.0 -m state --state INVAL
iptables -A FORWARD -m state --state ESTABLISHED,RELATED -j ACCEPT

# Some funny rules but not in a classic Iptables sorry ...
# Limit ICMP
# iptables -A FORWARD -p icmp -m limit --limit 4/s -j ACCEPT
# Match string, a good simple method to block some VIRUS very quickly
# iptables -I FORWARD -j DROP -p tcp -s 0.0.0.0/0 -m string --string "cmd.exe"

# Block all MySQL connection just to be sure
iptables -A FORWARD -p tcp -s 0/0 -d 62.3.3.0/24 --dport 3306 -j DROP

# Linux Mail Server Rules

# Allow FTP-DATA ( 20 ) , FTP ( 21 ) , SSH ( 22 )
iptables -A FORWARD -p tcp -s 0.0.0.0/0 -d 62.3.3.27/32 --dport 20:22 -j ACCEPT

# Allow the Mail Server to connect to the outside
# Note: This is *not* needed for the previous connections
# (remember: stateful filtering) and could be removed.
iptables -A FORWARD -p tcp -s 62.3.3.27/32 -d 0/0 -j ACCEPT

# WWW Server Rules

# Allow HTTP ( 80 ) connections with the WWW server
iptables -A FORWARD -p tcp -s 0.0.0.0/0 -d 62.3.3.28/32 --dport 80 -j ACCEPT

# Allow HTTPS ( 443 ) connections with the WWW server
iptables -A FORWARD -p tcp -s 0.0.0.0/0 -d 62.3.3.28/32 --dport 443 -j ACCEPT

# Allow the WWW server to go out
# Note: This is *not* needed for the previous connections
# (remember: stateful filtering) and could be removed.
iptables -A FORWARD -p tcp -s 62.3.3.28/32 -d 0/0 -j ACCEPT
```

Exemplo de script para alterar a instalação padrão do Bind.

This script automates the procedure for changing the **bind** version 8 name server's default installation so that it does *not* run as the superuser. Notice that **bind** version 9 in Debian already does this by default [3] , and you are much better using that version than **bind** version 8.

[3] Since version 9.2.1-5. That is, since Debian release *sarge*.

This script is here for historical purposes and to show how you can automate this kind of changes system-wide. The script will create the user and groups defined for the name server and will modify both `/etc/default/bind` and `/etc/init.d/bind` so that the program will run with that user. Use with extreme care since it has not been tested thoroughly.

You can also create the users manually and use the patch available for the default init.d script attached to http://bugs.debian.org/cgi-bin/bugreport.cgi?bug=157245.

```
  #!/bin/sh
  # Change the default Debian bind configuration to have it run
  # with a non-root user and group.
  #
  # WARN: This script has not been tested thoroughly, please
  # verify the changes made to the INITD script

  # (c) 2002 Javier Fernandez-Sanguino Peña
  #
  #    This program is free software; you can redistribute it and/or modify
  #    it under the terms of the GNU General Public License as published by
  #    the Free Software Foundation; either version 1, or (at your option)
  #    any later version.
  #
  #    This program is distributed in the hope that it will be useful,
  #    but WITHOUT ANY WARRANTY; without even the implied warranty of
  #    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
  #    GNU General Public License for more details.
  #
  #     Please see the file `COPYING' for the complete copyright notice.
  #

  restore() {
  # Just in case, restore the system if the changes fail
\t  echo "WARN: Restoring to the previous setup since I'm unable to properly chang
\t  echo "WARN: Please check the $INITDERR script."
\t  mv $INITD $INITDERR
\t  cp $INITDBAK $INITD
  }


  USER=named
  GROUP=named
  INITD=/etc/init.d/bind
  INITDBAK=$INITD.preuserchange
  INITDERR=$INITD.changeerror
  START="start-stop-daemon --start --quiet --exec /usr/sbin/named -- -g $GROUP -u
  AWKS="awk ' /start-stop-daemon --start/ { print \"$START\"; noprint = 1; }; /\/u

  [ `id -u` -ne 0 ] && {
\t  echo "This program must be run by the root user"
\t  exit 1
  }

  RUNUSER=`ps -eo user,fname |grep named |cut -f 1 -d " "`
```

```
  if [ "$RUNUSER" = "$USER" ]
  then
\t  echo "WARN: The name server running daemon is already running as $USER"
\t  echo "ERR:  This script will not many any changes to your setup."
\t  exit 1
  fi
  if [ ! -f $INITD ]
  then
\t  echo "ERR:  This system does not have $INITD (which this script tries to chang
\t  RUNNING=`ps -eo fname |grep named`
\t   [ -z "$RUNNING" ] && \
\t      echo "ERR:  In fact the name server daemon is not even running (is it inst
\t   echo "ERR:  No changes will be made to your system"
\t  exit 1
  fi

  # Check if named group exists
  if [ -z "`grep $GROUP /etc/group`" ]
  then
\t  echo "Creating group $GROUP:"
\t  addgroup $GROUP
  else
\t  echo "WARN: Group $GROUP already exists. Will not create it"
  fi
  # Same for the user
  if [ -z "`grep $USER /etc/passwd`" ]
  then
\t  echo "Creating user $USER:"
\t  adduser --system --home /home/$USER \
\t  --no-create-home --ingroup $GROUP \
\t  --disabled-password --disabled-login $USER
  else
\t  echo "WARN: The user $USER already exists. Will not create it"
  fi

  # Change the init.d script

  # First make a backup (check that there is not already
  # one there first)
  if [ ! -f $INITDBAK ]
  then
\t  cp $INITD $INITDBAK
  fi

  # Then use it to change it
  cat $INITDBAK |
  eval $AWKS > $INITD

  echo "WARN: The script $INITD has been changed, trying to test the changes."
  echo "Restarting the named daemon (check for errors here)."

  $INITD restart
  if [ $? -ne 0 ]
```

```
  then
\t  echo "ERR:  Failed to restart the daemon."
\t  restore
\t  exit 1
  fi

  RUNNING=`ps -eo fname |grep named`
  if [ -z "$RUNNING" ]
  then
\t  echo "ERR:  Named is not running, probably due to a problem with the changes."
\t  restore
\t  exit 1
  fi

  # Check if it's running as expected
  RUNUSER=`ps -eo user,fname |grep named |cut -f 1 -d " "`

  if [ "$RUNUSER" = "$USER" ]
  then
\t  echo "All has gone well, named seems to be running now as $USER."
  else
\t  echo "ERR:  The script failed to automatically change the system."
\t  echo "ERR:  Named is currently running as $RUNUSER."
\t  restore
\t  exit 1
  fi

  exit 0
```

O script anterior, execute-o no **bind** customizado do Woody (Debian 3.0), irá produzir o arquivo initd abaixo depois de criar o usuário e grupo 'named':

Atualização de segurança protegida por um firewall

Depois de uma instalaçõa padrão, o sistema ainda poderá ter algumas vulnerabilidades de segurança. Ao menos que você baixe as atualizações para os pacotes vulneráveis em outro computador (ou você tenha espelhado security.debian.org para uso local), o sistema deverá ter acesso à Internet para os downloads.

Entretanto, na medida que você se conecta à Internet estará expondo seu sistema. Se um de seus serviços locais estiver vulnerável, poderá ser comprometido mesmo antes de finalizar as atualizações! Isso pode ser paranóico, mas as análises do http://www.honeynet.org têm mostrado que sistemas podem ser comprometidos em menos de três dias, mesmo que o sistema não seja conhecido publicamento (i.e., não está publicado nos registros DNS).

Quando estiver fazendo uma atualização em um sistema não protegido por um mecanismo externo como firewall, é possível configurar seu firewall local para restringir conexões envolvendo somente as próprias atualizações de segurança. O exemplo abaixo mostra como configurar estas capacidades de firewall, que permitem somente conexões do security.debian.org, registrando todas as outras que são negadas.

The following example can be use to setup a restricted firewall ruleset. Run this commands from a local console (not a remote one) to reduce the chances of locking yourself out of the system.

```
# iptables -F
# iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source               destination

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
# iptables -P INPUT DROP
# iptables -P FORWARD DROP
# iptables -P OUTPUT DROP
# iptables -A OUTPUT -d security.debian.org --dport 80 -j ACCEPT
# iptables -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
# iptables -A INPUT -p icmp -j ACCEPT
# iptables -A INPUT -j LOG
# iptables -A OUTPUT -j LOG
# iptables -L
Chain INPUT (policy DROP)
target     prot opt source               destination
ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0          state RELATED,ESTABL
ACCEPT     icmp --  0.0.0.0/0            0.0.0.0/0
LOG        all  --  anywhere             anywhere           LOG level warning

Chain FORWARD (policy DROP)
target     prot opt source               destination

Chain OUTPUT (policy DROP)
target     prot opt source               destination
ACCEPT     80   --  anywhere             security.debian.org
LOG        all  --  anywhere             anywhere           LOG level warning
```

Note: Using a *DROP* policy in the INPUT chain is the most correct thing to do, but be *very* careful when doing this after flushing the chain from a remote connection. When testing firewall rulesets from a remote location it is best if you run a script with the firewall ruleset (instead of introducing the ruleset line by line through the command line) and, as a precaution, keep a backdoor[4]

[4] Such as *knockd*. Alternatively, you can open a different console and have the system ask for confirmation that there is somebody on the other side, and reset the firewall chain if no confirmation is given. The following test script could be of use:

```
#!/bin/bash

while true; do
    read -n 1 -p "Are you there? " -t 30 ayt
    if [ -z "$ayt" ] ; then
        break
    fi
done

# Reset the firewall chain, user is not available
echo
echo "Resetting firewall chain!"
iptables -F
iptables -P INPUT ACCEPT
iptables -P FORWARD ACCEPT
```

Of course, you should disable any backdoors before getting the system into production. configured so that you can re-enable access to the system if you make a mistake. That way there would be no need to go to a remote location to fix a firewall ruleset that blocks you.

FIXME: This needs DNS to be working properly since it is required for security.debian.org to work. You can add security.debian.org to /etc/hosts but now it is a CNAME to several hosts (there is more than one security mirror)

FIXME: this will only work with HTTP URLs since ftp might need the ip_conntrack_ftp module, or use passive mode.

Chroot environment for SSH

Creating a restricted environment for SSH is a tough job due to its dependencies and the fact that, unlike other servers, SSH provides a remote shell to users. Thus, you will also have to consider the applications users will be allowed to use in the environment.

You have two options to setup a restricted remote shell:

- Chrooting the ssh users, by properly configuring the ssh daemon you can ask it to chroot a user after authentication just before it is provided a shell. Each user can have their own environment.
- Chrooting the ssh server, since you chroot the ssh application itself all users are chrooted to the defined environment.

The first option has the advantage of making it possible to have both non-chrooted and chrooted users, if you don't introduce any setuid application in the user's chroots it is more difficult to break out of it. However, you might need to setup individual chroots for each user and it is more difficult to setup (as it requires cooperation from the SSH server). The second option is more easy to setup, and protects from an exploitation of the ssh server itself (since it's also in the chroot) but it will have the limitation that all users will share the same chroot environment (you cannot setup a per-user chroot environment).

Chrooting the ssh users

You can setup the ssh server so that it will chroot a set of defined users into a shell with a limited set of applications available.

Using libpam-chroot

Probably the easiest way is to use the libpam-chroot package provided in Debian. Once you install it you need to:

- Modify `/etc/pam.d/ssh` to use this PAM module, add as its last line[5]:

```
session    required   pam_chroot.so
```

```
iptables -P OUTPUT ACCEPT
exit 1
```

[5] You can use the *debug* option to have it send the progress of the module to the *authpriv.notice* facility

- set a proper chroot environment for the user. You can try using the scripts available at `/usr/share/doc/libpam-chroot/examples/`, use the makejail[6] program or setup a minimum Debian environment with debootstrap. Make sure the environment includes the needed devices [7].

- Configure `/etc/security/chroot.conf` so that the users you determine are chrooted to the directory you setup previously. You might want to have independent directories for different users so that they will not be able to see neither the whole system nor each other's.

- Configure SSH: Depending on your OpenSSH version the chroot environment might work straight of the box or not. Since 3.6.1p2 the *do_pam_session()* function is called after sshd has dropped privileges, since chroot() needs root priviledges it will not work with Privilege separation on. In newer OpenSSH versions, however, the PAM code has been modified and do_pam_session is called before dropping priviledges so it will work even with Privilege separation is on. If you have to disable it modify `/etc/ssh/sshd_config` like this:

  ```
  UsePrivilegeSeparation no
  ```

  Notice that this will lower the security of your system since the OpenSSH server will then run as *root* user. This means that if a remote attack is found against OpenSSH an attacker will get *root* privileges instead of *sshd*, thus compromising the whole system. [8]

If you don't disable *Privilege Separation* you will need an `/etc/passwd` which includes the user's UID inside the chroot for *Privilege Separation* to work properly.

If you have *Privilege Separation* set to *yes* and your OpenSSH version does not behave properly you will need to disable it. If you don't, users that try to connect to your server and would be chrooted by this module will see this:

```
$ ssh -l user server
user@server's password:
Connection to server closed by remote host.
Connection to server closed.
```

This is because the ssh daemon, which is running as 'sshd', is not be able to make the chroot() system call. To disable Privilege separation you have to modify the `/etc/ssh/sshd_config` configuration file as described above.

Notice that if any of the following is missing the users will not be able to logon to the chroot:

[6] You can create a very limited bash environment with the following python definition for makejail, just create the directory `/var/chroots/users/foo` and a file with the following contents and call it `bash.py`:

```
chroot="/var/chroots/users/foo"
cleanJailFirst=1
testCommandsInsideJail=["bash ls"]
```

And then run *makejail bash.py* to create the user environment at `/var/chroots/users/foo`. To test the environment run:

```
# chroot /var/chroots/users/foo/ ls
bin  dev  etc  lib  proc  sbin  usr
```

[7] In some occasions you might need the `/dev/ptmx` and `/dev/pty*` devices and the `/dev/pts/` subdirectory. Running MAKEDEV in the `/dev` directory of the chrooted environment should be sufficient to create them if they do not exist. If you are using kernels (version 2.6) which dynamically create device files you will need to create the /dev/pts/ files yourself and grant them the proper privileges.

[8] If you are using a kernel that implements Mandatory Access Control (RSBAC/SElinux) you can avoid changing this configuration just by granting the *sshd* user privileges to make the chroot() system call.

- The `/proc` filesystem needs to be mounted in the users' chroot.
- The necessary `/dev/pts/` devices need to exist. If the files are generated by your running kernel automatically then you have to manually create them on the chroot's `/dev/`.
- The user's home directory has to exist in the chroot, otherwise the ssh daemon will not continue.

You can debug all these issues if you use the *debug* keyword in the `/etc/pam.d/ssh` PAM definition. If you encounter issues you might find it useful to enable the debugging mode on the ssh client too.

Note: This information is also available (and maybe more up to date) in `/usr/share/doc/libpam-chroot/README.Debian.gz`, please review it for updated information before taking the above steps.

Patching the ssh server

Debian's **sshd** does not allow restriction of a user's movement through the server, since it lacks the **chroot** function that the commercial program **sshd2** includes (using 'ChrootGroups' or 'ChrootUsers', see sshd2_config(5)). However, there is a patch available to add this functionality available from http://chrootssh.sourceforge.net (requested and available in http://bugs.debian.org/139047 in Debian). The patch may be included in future releases of the OpenSSH package. Emmanuel Lacour has **ssh** deb packages for *sarge* with this feature. They are available at http://debian.home-dn.net/sarge/ssh/. Notice that those might not be up to date so completing the compilation step is recommended.

After applying the patch, modify `/etc/passwd` by changing the home path of the users (with the special `/./` token):

```
joeuser:x:1099:1099:Joe Random User:/home/joe/./:/bin/bash
```

Isto irá restringir *ambos* o acesso remoto ao shell, como também a cópia remota através do canal **ssh**.

Tenha certeza de ter todos os binários e bibliotecas necessárias dentro do caminho que está enjaulado para os usuários. Estes arquivos devem pertencer ao root para evitar tampering pelo usuário (como sair da jaula **chroot**'ed). Um exemplo possível inclue:

```
./bin:
total 660
drwxr-xr-x    2 root     root         4096 Mar 18 13:36 .
drwxr-xr-x    8 guest    guest        4096 Mar 15 16:53 ..
-r-xr-xr-x    1 root     root       531160 Feb  6 22:36 bash
-r-xr-xr-x    1 root     root        43916 Nov 29 13:19 ls
-r-xr-xr-x    1 root     root        16684 Nov 29 13:19 mkdir
-rwxr-xr-x    1 root     root        23960 Mar 18 13:36 more
-r-xr-xr-x    1 root     root         9916 Jul 26  2001 pwd
-r-xr-xr-x    1 root     root        24780 Nov 29 13:19 rm
lrwxrwxrwx    1 root     root            4 Mar 30 16:29 sh -> bash


./etc:
total 24
drwxr-xr-x    2 root     root         4096 Mar 15 16:13 .
drwxr-xr-x    8 guest    guest        4096 Mar 15 16:53 ..
-rw-r--r--    1 root     root           54 Mar 15 13:23 group
-rw-r--r--    1 root     root          428 Mar 15 15:56 hosts
-rw-r--r--    1 root     root           44 Mar 15 15:53 passwd
-rw-r--r--    1 root     root           52 Mar 15 13:23 shells
```

```
./lib:
total 1848
drwxr-xr-x    2 root     root         4096 Mar 18 13:37 .
drwxr-xr-x    8 guest    guest        4096 Mar 15 16:53 ..
-rwxr-xr-x    1 root     root        92511 Mar 15 12:49 ld-linux.so.2
-rwxr-xr-x    1 root     root      1170812 Mar 15 12:49 libc.so.6
-rw-r--r--    1 root     root        20900 Mar 15 13:01 libcrypt.so.1
-rw-r--r--    1 root     root         9436 Mar 15 12:49 libdl.so.2
-rw-r--r--    1 root     root       248132 Mar 15 12:48 libncurses.so.5
-rw-r--r--    1 root     root        71332 Mar 15 13:00 libnsl.so.1
-rw-r--r--    1 root     root        34144 Mar 15 16:10
libnss_files.so.2
-rw-r--r--    1 root     root        29420 Mar 15 12:57 libpam.so.0
-rw-r--r--    1 root     root       105498 Mar 15 12:51 libpthread.so.0
-rw-r--r--    1 root     root        25596 Mar 15 12:51 librt.so.1
-rw-r--r--    1 root     root         7760 Mar 15 12:59 libutil.so.1
-rw-r--r--    1 root     root        24328 Mar 15 12:57 libwrap.so.0


./usr:
total 16
drwxr-xr-x    4 root     root         4096 Mar 15 13:00 .
drwxr-xr-x    8 guest    guest        4096 Mar 15 16:53 ..
drwxr-xr-x    2 root     root         4096 Mar 15 15:55 bin
drwxr-xr-x    2 root     root         4096 Mar 15 15:37 lib


./usr/bin:
total 340
drwxr-xr-x    2 root     root         4096 Mar 15 15:55 .
drwxr-xr-x    4 root     root         4096 Mar 15 13:00 ..
-rwxr-xr-x    1 root     root        10332 Mar 15 15:55 env
-rwxr-xr-x    1 root     root        13052 Mar 15 13:13 id
-r-xr-xr-x    1 root     root        25432 Mar 15 12:40 scp
-rwxr-xr-x    1 root     root        43768 Mar 15 15:15 sftp
-r-sr-xr-x    1 root     root       218456 Mar 15 12:40 ssh
-rwxr-xr-x    1 root     root         9692 Mar 15 13:17 tty


./usr/lib:
total 852
drwxr-xr-x    2 root     root         4096 Mar 15 15:37 .
drwxr-xr-x    4 root     root         4096 Mar 15 13:00 ..
-rw-r--r--    1 root     root       771088 Mar 15 13:01
libcrypto.so.0.9.6
-rw-r--r--    1 root     root        54548 Mar 15 13:00 libz.so.1
-rwxr-xr-x    1 root     root        23096 Mar 15 15:37 sftp-server
```

Chrooting the ssh server

If you create a chroot which includes the SSH server files in, for example `/var/chroot/ssh`, you would start the **ssh** server **chroot**'ed with this command:

```
# chroot /var/chroot/ssh /sbin/sshd -f /etc/sshd_config
```

That would make startup the **sshd** daemon inside the chroot. In order to do that you have to first prepare the contents of the `/var/chroot/ssh` directory so that it includes both the SSH server and all the utilities that the users connecting to that server might need. If you are doing this you should make certain that OpenSSH uses *Privilege Separation* (which is the default) having the following line in the configuration file `/etc/ssh/sshd_config`:

```
UsePrivilegeSeparation yes
```

That way the remote daemon will do as few things as possible as the root user so even if there is a bug in it it will not compromise the chroot. Notice that, unlike the case in which you setup a per-user chroot, the ssh daemon is running in the same chroot as the users so there is at least one potential process running as root which could break out of the chroot.

Notice, also, that in order for SSH to work in that location, the partition where the chroot directory resides cannot be mounted with the *nodev* option. If you use that option, then you will get the following error: *PRNG is not seeded*, because `/dev/urandom` does not work in the chroot.

Setup a minimal system (the really easy way)

You can use debootstrap to setup a minimal environment that just includes the ssh server. In order to do this you just have to create a chroot as described in the http://www.debian.org/doc/manuals/reference/ch09#_chroot_system document. This method is bound to work (you will get all the necessary componentes for the chroot) but at the cost of disk space (a minimal installation of Debian will amount to several hundred megabytes). This minimal system might also include setuid files that a user in the chroot could use to break out of the chroot if any of those could be use for a privilege escalation.

Configurando automaticamente o ambiente (a maneira fácil)

Você pode facilmente criar um ambiente restrito com o pacote makejail, já que ele automaticamente segue as trilhas do servidor daemon (com **strace**) e faz com que ele execute em um ambiente restrito.

A vantagem de programas que automaticamente geram um ambiente **chroot** é que eles são capazes de copiar qualquer pacote para o ambiente **chroot** (mesmo seguindo as dependências do pacote e certificar que foi completada). Então, fornecer as aplicações dos usuários é bem mais fácil.

To set up the environment using **makejail**'s provided examples, just create `/var/chroot/sshd` and use the command:

```
# makejail /usr/share/doc/makejail/examples/sshd.py
```

This will setup the chroot in the `/var/chroot/sshd` directory. Notice that this chroot will not fully work unless you:

- Mount the *procfs* filesystem in `/var/chroot/sshd/proc`. **Makejail** will mount it for you but if the system reboots you need to remount it running:

  ```
  # mount -t proc proc /var/chroot/sshd/proc
  ```

  You can also have it be mounted automatically by editing `/etc/fstab` and including this line:

```
proc-ssh /var/chroot/sshd/proc  proc none 0 0
```

- Have syslog listen to the device `/dev/log` inside the chroot. In order to do this you have modify `/etc/default/syslogd` and add *-a /var/chroot/sshd/dev/log* to the `SYSLOGD` variable definition.

Leia o arquivo exemplo para ver que outras mudanças devem ser feitas para o ambiente. Algumas dessas mudanças, como copiar os diretórios home do usuário, não podem ser feitas automaticamente. Também limite a exposição de informações sensíveis, copiando os dados de um certo número de usuários dos arquivos `/etc/shadow` ou `/etc/group`.

O seguinte exemplo de ambiente tem sido (levemente) testado, foi construído com o arquivo de configuração fornecido no pacote e inclue o pacote fileutils:

```
.
|-- bin
|   |-- ash
|   |-- bash
|   |-- chgrp
|   |-- chmod
|   |-- chown
|   |-- cp
|   |-- csh -> /etc/alternatives/csh
|   |-- dd
|   |-- df
|   |-- dir
|   |-- fdflush
|   |-- ksh
|   |-- ln
|   |-- ls
|   |-- mkdir
|   |-- mknod
|   |-- mv
|   |-- rbash -> bash
|   |-- rm
|   |-- rmdir
|   |-- sh -> bash
|   |-- sync
|   |-- tcsh
|   |-- touch
|   |-- vdir
|   |-- zsh -> /etc/alternatives/zsh
|   `-- zsh4
|-- dev
|   |-- null
|   |-- ptmx
|   |-- pts
|   |-- ptya0
(...)
|   |-- tty
|   |-- tty0
(...)
|   `-- urandom
|-- etc
|   |-- alternatives
```

```
|   |   |-- csh -> /bin/tcsh
|   |   `-- zsh -> /bin/zsh4
|   |-- environment
|   |-- hosts
|   |-- hosts.allow
|   |-- hosts.deny
|   |-- ld.so.conf
|   |-- localtime -> /usr/share/zoneinfo/Europe/Madrid
|   |-- motd
|   |-- nsswitch.conf
|   |-- pam.conf
|   |-- pam.d
|   |   |-- other
|   |   `-- ssh
|   |-- passwd
|   |-- resolv.conf
|   |-- security
|   |   |-- access.conf
|   |   |-- chroot.conf
|   |   |-- group.conf
|   |   |-- limits.conf
|   |   |-- pam_env.conf
|   |   `-- time.conf
|   |-- shadow
|   |-- shells
|   `-- ssh
|       |-- moduli
|       |-- ssh_host_dsa_key
|       |-- ssh_host_dsa_key.pub
|       |-- ssh_host_rsa_key
|       |-- ssh_host_rsa_key.pub
|       `-- sshd_config
|-- home
|   `-- userX
|-- lib
|   |-- ld-2.2.5.so
|   |-- ld-linux.so.2 -> ld-2.2.5.so
|   |-- libc-2.2.5.so
|   |-- libc.so.6 -> libc-2.2.5.so
|   |-- libcap.so.1 -> libcap.so.1.10
|   |-- libcap.so.1.10
|   |-- libcrypt-2.2.5.so
|   |-- libcrypt.so.1 -> libcrypt-2.2.5.so
|   |-- libdl-2.2.5.so
|   |-- libdl.so.2 -> libdl-2.2.5.so
|   |-- libm-2.2.5.so
|   |-- libm.so.6 -> libm-2.2.5.so
|   |-- libncurses.so.5 -> libncurses.so.5.2
|   |-- libncurses.so.5.2
|   |-- libnsl-2.2.5.so
|   |-- libnsl.so.1 -> libnsl-2.2.5.so
|   |-- libnss_compat-2.2.5.so
|   |-- libnss_compat.so.2 -> libnss_compat-2.2.5.so
|   |-- libnss_db-2.2.so
```

```
|   |-- libnss_db.so.2 -> libnss_db-2.2.so
|   |-- libnss_dns-2.2.5.so
|   |-- libnss_dns.so.2 -> libnss_dns-2.2.5.so
|   |-- libnss_files-2.2.5.so
|   |-- libnss_files.so.2 -> libnss_files-2.2.5.so
|   |-- libnss_hesiod-2.2.5.so
|   |-- libnss_hesiod.so.2 -> libnss_hesiod-2.2.5.so
|   |-- libnss_nis-2.2.5.so
|   |-- libnss_nis.so.2 -> libnss_nis-2.2.5.so
|   |-- libnss_nisplus-2.2.5.so
|   |-- libnss_nisplus.so.2 -> libnss_nisplus-2.2.5.so
|   |-- libpam.so.0 -> libpam.so.0.72
|   |-- libpam.so.0.72
|   |-- libpthread-0.9.so
|   |-- libpthread.so.0 -> libpthread-0.9.so
|   |-- libresolv-2.2.5.so
|   |-- libresolv.so.2 -> libresolv-2.2.5.so
|   |-- librt-2.2.5.so
|   |-- librt.so.1 -> librt-2.2.5.so
|   |-- libutil-2.2.5.so
|   |-- libutil.so.1 -> libutil-2.2.5.so
|   |-- libwrap.so.0 -> libwrap.so.0.7.6
|   |-- libwrap.so.0.7.6
|   `-- security
|       |-- pam_access.so
|       |-- pam_chroot.so
|       |-- pam_deny.so
|       |-- pam_env.so
|       |-- pam_filter.so
|       |-- pam_ftp.so
|       |-- pam_group.so
|       |-- pam_issue.so
|       |-- pam_lastlog.so
|       |-- pam_limits.so
|       |-- pam_listfile.so
|       |-- pam_mail.so
|       |-- pam_mkhomedir.so
|       |-- pam_motd.so
|       |-- pam_nologin.so
|       |-- pam_permit.so
|       |-- pam_rhosts_auth.so
|       |-- pam_rootok.so
|       |-- pam_securetty.so
|       |-- pam_shells.so
|       |-- pam_stress.so
|       |-- pam_tally.so
|       |-- pam_time.so
|       |-- pam_unix.so
|       |-- pam_unix_acct.so -> pam_unix.so
|       |-- pam_unix_auth.so -> pam_unix.so
|       |-- pam_unix_passwd.so -> pam_unix.so
|       |-- pam_unix_session.so -> pam_unix.so
|       |-- pam_userdb.so
|       |-- pam_warn.so
```

```
|       `-- pam_wheel.so
|-- sbin
|   `-- start-stop-daemon
|-- usr
|   |-- bin
|   |   |-- dircolors
|   |   |-- du
|   |   |-- install
|   |   |-- link
|   |   |-- mkfifo
|   |   |-- shred
|   |   |-- touch -> /bin/touch
|   |   `-- unlink
|   |-- lib
|   |   |-- libcrypto.so.0.9.6
|   |   |-- libdb3.so.3 -> libdb3.so.3.0.2
|   |   |-- libdb3.so.3.0.2
|   |   |-- libz.so.1 -> libz.so.1.1.4
|   |   `-- libz.so.1.1.4
|   |-- sbin
|   |   `-- sshd
|   `-- share
|       |-- locale
|       |   `-- es
|       |       |-- LC_MESSAGES
|       |       |   |-- fileutils.mo
|       |       |   |-- libc.mo
|       |       |   `-- sh-utils.mo
|       |       `-- LC_TIME -> LC_MESSAGES
|       `-- zoneinfo
|           `-- Europe
|               `-- Madrid
`-- var
    `-- run
        |-- sshd
        `-- sshd.pid

27 directories, 733 files
```

For Debian release 3.1 you have to make sure that the environment includes also the common files for PAM. The following files need to be copied over to the chroot if **makejail** did not do it for you:

```
$ ls /etc/pam.d/common-*
/etc/pam.d/common-account  /etc/pam.d/common-password
/etc/pam.d/common-auth     /etc/pam.d/common-session
```

Manually creating the environment (the hard way)

É possível criar um ambiente, usando o método de tentativa e erro, seguindo a execução do servidor **sshd** e arquivos de log para determinar os arquivos necessários. O seguinte ambiente, contribuído por José Luis Ledesma, é uma listagem amostral do arquivos que estão no ambiente **chroot** para o **ssh**: [9]

[9] Observe que não existem arquivos SETUID. Isso torna mais difícil para usuários remotos fugir o ambiente **chroot**. Entretanto, isso também previne que os usuários alterem suas senhas, já que o programa **passwd** não pode modificar os arquivos `/etc/passwd` ou `/etc/shadow`.

```
.:
total 36
drwxr-xr-x 9 root root 4096 Jun 5 10:05 ./
drwxr-xr-x 11 root root 4096 Jun 3 13:43 ../
drwxr-xr-x 2 root root 4096 Jun 4 12:13 bin/
drwxr-xr-x 2 root root 4096 Jun 4 12:16 dev/
drwxr-xr-x 4 root root 4096 Jun 4 12:35 etc/
drwxr-xr-x 3 root root 4096 Jun 4 12:13 lib/
drwxr-xr-x 2 root root 4096 Jun 4 12:35 sbin/
drwxr-xr-x 2 root root 4096 Jun 4 12:32 tmp/
drwxr-xr-x 2 root root 4096 Jun 4 12:16 usr/
./bin:
total 8368
drwxr-xr-x 2 root root 4096 Jun 4 12:13 ./
drwxr-xr-x 9 root root 4096 Jun 5 10:05 ../
-rwxr-xr-x 1 root root 109855 Jun 3 13:45 a2p*
-rwxr-xr-x 1 root root 387764 Jun 3 13:45 bash*
-rwxr-xr-x 1 root root 36365 Jun 3 13:45 c2ph*
-rwxr-xr-x 1 root root 20629 Jun 3 13:45 dprofpp*
-rwxr-xr-x 1 root root 6956 Jun 3 13:46 env*
-rwxr-xr-x 1 root root 158116 Jun 3 13:45 fax2ps*
-rwxr-xr-x 1 root root 104008 Jun 3 13:45 faxalter*
-rwxr-xr-x 1 root root 89340 Jun 3 13:45 faxcover*
-rwxr-xr-x 1 root root 441584 Jun 3 13:45 faxmail*
-rwxr-xr-x 1 root root 96036 Jun 3 13:45 faxrm*
-rwxr-xr-x 1 root root 107000 Jun 3 13:45 faxstat*
-rwxr-xr-x 1 root root 77832 Jun 4 11:46 grep*
-rwxr-xr-x 1 root root 19597 Jun 3 13:45 h2ph*
-rwxr-xr-x 1 root root 46979 Jun 3 13:45 h2xs*
-rwxr-xr-x 1 root root 10420 Jun 3 13:46 id*
-rwxr-xr-x 1 root root 4528 Jun 3 13:46 ldd*
-rwxr-xr-x 1 root root 111386 Jun 4 11:46 less*
-r-xr-xr-x 1 root root 26168 Jun 3 13:45 login*
-rwxr-xr-x 1 root root 49164 Jun 3 13:45 ls*
-rwxr-xr-x 1 root root 11600 Jun 3 13:45 mkdir*
-rwxr-xr-x 1 root root 24780 Jun 3 13:45 more*
-rwxr-xr-x 1 root root 154980 Jun 3 13:45 pal2rgb*
-rwxr-xr-x 1 root root 27920 Jun 3 13:46 passwd*
-rwxr-xr-x 1 root root 4241 Jun 3 13:45 pl2pm*
-rwxr-xr-x 1 root root 2350 Jun 3 13:45 pod2html*
-rwxr-xr-x 1 root root 7875 Jun 3 13:45 pod2latex*
-rwxr-xr-x 1 root root 17587 Jun 3 13:45 pod2man*
-rwxr-xr-x 1 root root 6877 Jun 3 13:45 pod2text*
-rwxr-xr-x 1 root root 3300 Jun 3 13:45 pod2usage*
-rwxr-xr-x 1 root root 3341 Jun 3 13:45 podchecker*
-rwxr-xr-x 1 root root 2483 Jun 3 13:45 podselect*
-r-xr-xr-x 1 root root 82412 Jun 4 11:46 ps*
-rwxr-xr-x 1 root root 36365 Jun 3 13:45 pstruct*
-rwxr-xr-x 1 root root 7120 Jun 3 13:45 pwd*
-rwxr-xr-x 1 root root 179884 Jun 3 13:45 rgb2ycbcr*
-rwxr-xr-x 1 root root 20532 Jun 3 13:45 rm*
-rwxr-xr-x 1 root root 6720 Jun 4 10:15 rmdir*
-rwxr-xr-x 1 root root 14705 Jun 3 13:45 s2p*
```

```
-rwxr-xr-x 1 root root 28764 Jun 3 13:46 scp*
-rwxr-xr-x 1 root root 385000 Jun 3 13:45 sendfax*
-rwxr-xr-x 1 root root 67548 Jun 3 13:45 sendpage*
-rwxr-xr-x 1 root root 88632 Jun 3 13:46 sftp*
-rwxr-xr-x 1 root root 387764 Jun 3 13:45 sh*
-rws--x--x 1 root root 744500 Jun 3 13:46 slogin*
-rwxr-xr-x 1 root root 14523 Jun 3 13:46 splain*
-rws--x--x 1 root root 744500 Jun 3 13:46 ssh*
-rwxr-xr-x 1 root root 570960 Jun 3 13:46 ssh-add*
-rwxr-xr-x 1 root root 502952 Jun 3 13:46 ssh-agent*
-rwxr-xr-x 1 root root 575740 Jun 3 13:46 ssh-keygen*
-rwxr-xr-x 1 root root 383480 Jun 3 13:46 ssh-keyscan*
-rwxr-xr-x 1 root root 39 Jun 3 13:46 ssh_europa*
-rwxr-xr-x 1 root root 107252 Jun 4 10:14 strace*
-rwxr-xr-x 1 root root 8323 Jun 4 10:14 strace-graph*
-rwxr-xr-x 1 root root 158088 Jun 3 13:46 thumbnail*
-rwxr-xr-x 1 root root 6312 Jun 3 13:46 tty*
-rwxr-xr-x 1 root root 55904 Jun 4 11:46 useradd*
-rwxr-xr-x 1 root root 585656 Jun 4 11:47 vi*
-rwxr-xr-x 1 root root 6444 Jun 4 11:45 whoami*
./dev:
total 8
drwxr-xr-x 2 root root 4096 Jun 4 12:16 ./
drwxr-xr-x 9 root root 4096 Jun 5 10:05 ../
crw-r--r-- 1 root root 1, 9 Jun 3 13:43 urandom
./etc:
total 208
drwxr-xr-x 4 root root 4096 Jun 4 12:35 ./
drwxr-xr-x 9 root root 4096 Jun 5 10:05 ../
-rw------- 1 root root 0 Jun 4 11:46 .pwd.lock
-rw-r--r-- 1 root root 653 Jun 3 13:46 group
-rw-r--r-- 1 root root 242 Jun 4 11:33 host.conf
-rw-r--r-- 1 root root 857 Jun 4 12:04 hosts
-rw-r--r-- 1 root root 1050 Jun 4 11:29 ld.so.cache
-rw-r--r-- 1 root root 304 Jun 4 11:28 ld.so.conf
-rw-r--r-- 1 root root 235 Jun 4 11:27 ld.so.conf~
-rw-r--r-- 1 root root 88039 Jun 3 13:46 moduli
-rw-r--r-- 1 root root 1342 Jun 4 11:34 nsswitch.conf
drwxr-xr-x 2 root root 4096 Jun 4 12:02 pam.d/
-rw-r--r-- 1 root root 28 Jun 4 12:00 pam_smb.conf
-rw-r--r-- 1 root root 2520 Jun 4 11:57 passwd
-rw-r--r-- 1 root root 7228 Jun 3 13:48 profile
-rw-r--r-- 1 root root 1339 Jun 4 11:33 protocols
-rw-r--r-- 1 root root 274 Jun 4 11:44 resolv.conf
drwxr-xr-x 2 root root 4096 Jun 3 13:43 security/
-rw-r----- 1 root root 1178 Jun 4 11:51 shadow
-rw------- 1 root root 80 Jun 4 11:45 shadow-
-rw-r----- 1 root root 1178 Jun 4 11:48 shadow.old
-rw-r--r-- 1 root root 161 Jun 3 13:46 shells
-rw-r--r-- 1 root root 1144 Jun 3 13:46 ssh_config
-rw------- 1 root root 668 Jun 3 13:46 ssh_host_dsa_key
-rw-r--r-- 1 root root 602 Jun 3 13:46 ssh_host_dsa_key.pub
-rw------- 1 root root 527 Jun 3 13:46 ssh_host_key
-rw-r--r-- 1 root root 331 Jun 3 13:46 ssh_host_key.pub
```

```
-rw------- 1 root root 883 Jun 3 13:46 ssh_host_rsa_key
-rw-r--r-- 1 root root 222 Jun 3 13:46 ssh_host_rsa_key.pub
-rw-r--r-- 1 root root 2471 Jun 4 12:15 sshd_config
./etc/pam.d:
total 24
drwxr-xr-x 2 root root 4096 Jun 4 12:02 ./
drwxr-xr-x 4 root root 4096 Jun 4 12:35 ../
lrwxrwxrwx 1 root root 4 Jun 4 12:02 other -> sshd
-rw-r--r-- 1 root root 318 Jun 3 13:46 passwd
-rw-r--r-- 1 root root 546 Jun 4 11:36 ssh
-rw-r--r-- 1 root root 479 Jun 4 12:02 sshd
-rw-r--r-- 1 root root 370 Jun 3 13:46 su
./etc/security:
total 32
drwxr-xr-x 2 root root 4096 Jun 3 13:43 ./
drwxr-xr-x 4 root root 4096 Jun 4 12:35 ../
-rw-r--r-- 1 root root 1971 Jun 3 13:46 access.conf
-rw-r--r-- 1 root root 184 Jun 3 13:46 chroot.conf
-rw-r--r-- 1 root root 2145 Jun 3 13:46 group.conf
-rw-r--r-- 1 root root 1356 Jun 3 13:46 limits.conf
-rw-r--r-- 1 root root 2858 Jun 3 13:46 pam_env.conf
-rw-r--r-- 1 root root 2154 Jun 3 13:46 time.conf
./lib:
total 8316
drwxr-xr-x 3 root root 4096 Jun 4 12:13 ./
drwxr-xr-x 9 root root 4096 Jun 5 10:05 ../
-rw-r--r-- 1 root root 1024 Jun 4 11:51 cracklib_dict.hwm
-rw-r--r-- 1 root root 214324 Jun 4 11:51 cracklib_dict.pwd
-rw-r--r-- 1 root root 11360 Jun 4 11:51 cracklib_dict.pwi
-rwxr-xr-x 1 root root 342427 Jun 3 13:46 ld-linux.so.2*
-rwxr-xr-x 1 root root 4061504 Jun 3 13:46 libc.so.6*
lrwxrwxrwx 1 root root 15 Jun 4 12:11 libcrack.so -> libcrack.so.2.7*
lrwxrwxrwx 1 root root 15 Jun 4 12:11 libcrack.so.2 -> libcrack.so.2.7*
-rwxr-xr-x 1 root root 33291 Jun 4 11:39 libcrack.so.2.7*
-rwxr-xr-x 1 root root 60988 Jun 3 13:46 libcrypt.so.1*
-rwxr-xr-x 1 root root 71846 Jun 3 13:46 libdl.so.2*
-rwxr-xr-x 1 root root 27762 Jun 3 13:46 libhistory.so.4.0*
lrwxrwxrwx 1 root root 17 Jun 4 12:12 libncurses.so.4 -> libncurses.so.4.2*
-rwxr-xr-x 1 root root 503903 Jun 3 13:46 libncurses.so.4.2*
lrwxrwxrwx 1 root root 17 Jun 4 12:12 libncurses.so.5 -> libncurses.so.5.0*
-rwxr-xr-x 1 root root 549429 Jun 3 13:46 libncurses.so.5.0*
-rwxr-xr-x 1 root root 369801 Jun 3 13:46 libnsl.so.1*
-rwxr-xr-x 1 root root 142563 Jun 4 11:49 libnss_compat.so.1*
-rwxr-xr-x 1 root root 215569 Jun 4 11:49 libnss_compat.so.2*
-rwxr-xr-x 1 root root 61648 Jun 4 11:34 libnss_dns.so.1*
-rwxr-xr-x 1 root root 63453 Jun 4 11:34 libnss_dns.so.2*
-rwxr-xr-x 1 root root 63782 Jun 4 11:34 libnss_dns6.so.2*
-rwxr-xr-x 1 root root 205715 Jun 3 13:46 libnss_files.so.1*
-rwxr-xr-x 1 root root 235932 Jun 3 13:49 libnss_files.so.2*
-rwxr-xr-x 1 root root 204383 Jun 4 11:33 libnss_nis.so.1*
-rwxr-xr-x 1 root root 254023 Jun 4 11:33 libnss_nis.so.2*
-rwxr-xr-x 1 root root 256465 Jun 4 11:33 libnss_nisplus.so.2*
lrwxrwxrwx 1 root root 14 Jun 4 12:12 libpam.so.0 -> libpam.so.0.72*
-rwxr-xr-x 1 root root 31449 Jun 3 13:46 libpam.so.0.72*
```

```
lrwxrwxrwx 1 root root 19 Jun 4 12:12 libpam_misc.so.0 ->
libpam_misc.so.0.72*
-rwxr-xr-x 1 root root 8125 Jun 3 13:46 libpam_misc.so.0.72*
lrwxrwxrwx 1 root root 15 Jun 4 12:12 libpamc.so.0 -> libpamc.so.0.72*
-rwxr-xr-x 1 root root 10499 Jun 3 13:46 libpamc.so.0.72*
-rwxr-xr-x 1 root root 176427 Jun 3 13:46 libreadline.so.4.0*
-rwxr-xr-x 1 root root 44729 Jun 3 13:46 libutil.so.1*
-rwxr-xr-x 1 root root 70254 Jun 3 13:46 libz.a*
lrwxrwxrwx 1 root root 13 Jun 4 12:13 libz.so -> libz.so.1.1.3*
lrwxrwxrwx 1 root root 13 Jun 4 12:13 libz.so.1 -> libz.so.1.1.3*
-rwxr-xr-x 1 root root 63312 Jun 3 13:46 libz.so.1.1.3*
drwxr-xr-x 2 root root 4096 Jun 4 12:00 security/
./lib/security:
total 668
drwxr-xr-x 2 root root 4096 Jun 4 12:00 ./
drwxr-xr-x 3 root root 4096 Jun 4 12:13 ../
-rwxr-xr-x 1 root root 10067 Jun 3 13:46 pam_access.so*
-rwxr-xr-x 1 root root 8300 Jun 3 13:46 pam_chroot.so*
-rwxr-xr-x 1 root root 14397 Jun 3 13:46 pam_cracklib.so*
-rwxr-xr-x 1 root root 5082 Jun 3 13:46 pam_deny.so*
-rwxr-xr-x 1 root root 13153 Jun 3 13:46 pam_env.so*
-rwxr-xr-x 1 root root 13371 Jun 3 13:46 pam_filter.so*
-rwxr-xr-x 1 root root 7957 Jun 3 13:46 pam_ftp.so*
-rwxr-xr-x 1 root root 12771 Jun 3 13:46 pam_group.so*
-rwxr-xr-x 1 root root 10174 Jun 3 13:46 pam_issue.so*
-rwxr-xr-x 1 root root 9774 Jun 3 13:46 pam_lastlog.so*
-rwxr-xr-x 1 root root 13591 Jun 3 13:46 pam_limits.so*
-rwxr-xr-x 1 root root 11268 Jun 3 13:46 pam_listfile.so*
-rwxr-xr-x 1 root root 11182 Jun 3 13:46 pam_mail.so*
-rwxr-xr-x 1 root root 5923 Jun 3 13:46 pam_nologin.so*
-rwxr-xr-x 1 root root 5460 Jun 3 13:46 pam_permit.so*
-rwxr-xr-x 1 root root 18226 Jun 3 13:46 pam_pwcheck.so*
-rwxr-xr-x 1 root root 12590 Jun 3 13:46 pam_rhosts_auth.so*
-rwxr-xr-x 1 root root 5551 Jun 3 13:46 pam_rootok.so*
-rwxr-xr-x 1 root root 7239 Jun 3 13:46 pam_securetty.so*
-rwxr-xr-x 1 root root 6551 Jun 3 13:46 pam_shells.so*
-rwxr-xr-x 1 root root 55925 Jun 4 12:00 pam_smb_auth.so*
-rwxr-xr-x 1 root root 12678 Jun 3 13:46 pam_stress.so*
-rwxr-xr-x 1 root root 11170 Jun 3 13:46 pam_tally.so*
-rwxr-xr-x 1 root root 11124 Jun 3 13:46 pam_time.so*
-rwxr-xr-x 1 root root 45703 Jun 3 13:46 pam_unix.so*
-rwxr-xr-x 1 root root 45703 Jun 3 13:46 pam_unix2.so*
-rwxr-xr-x 1 root root 45386 Jun 3 13:46 pam_unix_acct.so*
-rwxr-xr-x 1 root root 45386 Jun 3 13:46 pam_unix_auth.so*
-rwxr-xr-x 1 root root 45386 Jun 3 13:46 pam_unix_passwd.so*
-rwxr-xr-x 1 root root 45386 Jun 3 13:46 pam_unix_session.so*
-rwxr-xr-x 1 root root 9726 Jun 3 13:46 pam_userdb.so*
-rwxr-xr-x 1 root root 6424 Jun 3 13:46 pam_warn.so*
-rwxr-xr-x 1 root root 7460 Jun 3 13:46 pam_wheel.so*
./sbin:
total 3132
drwxr-xr-x 2 root root 4096 Jun 4 12:35 ./
drwxr-xr-x 9 root root 4096 Jun 5 10:05 ../
-rwxr-xr-x 1 root root 178256 Jun 3 13:46 choptest*
```

```
-rwxr-xr-x 1 root root 184032 Jun 3 13:46 cqtest*
-rwxr-xr-x 1 root root 81096 Jun 3 13:46 dialtest*
-rwxr-xr-x 1 root root 1142128 Jun 4 11:28 ldconfig*
-rwxr-xr-x 1 root root 2868 Jun 3 13:46 lockname*
-rwxr-xr-x 1 root root 3340 Jun 3 13:46 ondelay*
-rwxr-xr-x 1 root root 376796 Jun 3 13:46 pagesend*
-rwxr-xr-x 1 root root 13950 Jun 3 13:46 probemodem*
-rwxr-xr-x 1 root root 9234 Jun 3 13:46 recvstats*
-rwxr-xr-x 1 root root 64480 Jun 3 13:46 sftp-server*
-rwxr-xr-x 1 root root 744412 Jun 3 13:46 sshd*
-rwxr-xr-x 1 root root 30750 Jun 4 11:46 su*
-rwxr-xr-x 1 root root 194632 Jun 3 13:46 tagtest*
-rwxr-xr-x 1 root root 69892 Jun 3 13:46 tsitest*
-rwxr-xr-x 1 root root 43792 Jun 3 13:46 typetest*
./tmp:
total 8
drwxr-xr-x 2 root root 4096 Jun 4 12:32 ./
drwxr-xr-x 9 root root 4096 Jun 5 10:05 ../
./usr:
total 8
drwxr-xr-x 2 root root 4096 Jun 4 12:16 ./
drwxr-xr-x 9 root root 4096 Jun 5 10:05 ../
lrwxrwxrwx 1 root root 7 Jun 4 12:14 bin -> ../bin//
lrwxrwxrwx 1 root root 7 Jun 4 11:33 lib -> ../lib//
lrwxrwxrwx 1 root root 8 Jun 4 12:13 sbin -> ../sbin//
```

Chroot environment for Apache

Introduçao

O utilitário **chroot** é muitas vezes usado para enjaular um daemon dentro de uma estrutura restrita. Voce pode usá-lo para isolar um serviço do outro, desta forma um problema de segurança em um pacote de software específico não interfere em todo o servidor. A utilização do script **makejail** torna a configuração e atualização da árvore enjaulada muito mais fácil.

FIXME: Apache também pode ser enjaulado usando http://www.modsecurity.org que está disponível em libapache-mod-security (para Apache 1.x) e libapache2-mod-security (para Apache 2.x).

Licença

Instalando o servidor

Este procedimento foi testado no Debian GNU/Linux 3.0 (Woody) com **makejail** 0.0.4-1 (em Debian/testing).

- Efetue o login como **root** e crie um novo diretório para jaula:

```
$ mkdir -p /var/chroot/apache
```

- Crie um novo usuário e novo grupo. O servidor Apache enjaulado irá executar com este usuário/grupo, que não é utilizado para mais nada no sistema. Neste exemplo, ambos usuário e grupo são chamados de **chrapach**.

```
$ adduser --home /var/chroot/apache --shell /bin/false \
--no-create-home --system --group chrapach
```

  FIXME: é preciso um novo usuário? (Apache já executa como usuário apache)

- Instale o Apache normalmente no Debian: `apt-get install apache`

- Configure o Apache (por exemplo defina seus subdomínios e etc.). No arquivo de configuração `/etc/apache/httpd.conf`, altere as opções *Group* e *User* para `chrapach`. Reinicie o Apache e tenha certeza que o servidor está funcionando corretamente. Agora, pare o daemon do Apache.

- Instale o **makejail** (disponível agora no Debian/testing). Você também deve instalar **wget** e **lynx**, pois eles serão usados pelo **makejail** para testar o servidor enjaulado: `apt-get install makejail wget lynx`

- Copie o arquivo de configuração de exemplo para o Apache para o diretório `/etc/makejail`:

```
# cp /usr/share/doc/makejail/examples/apache.py /etc/makejail/
```

- Edit `/etc/makejail/apache.py`. You need to change the *chroot*, *users* and *groups* options. To run this version of **makejail**, you can also add a **packages** option. See the http://www.floc.net/makejail/current/doc/. A sample is shown here:

```
chroot="/var/chroot/apache"
testCommandsInsideJail=["/usr/sbin/apachectl start"]
processNames=["apache"]
testCommandsOutsideJail=["wget -r --spider http://localhost/",
                         "lynx --source https://localhost/"]
preserve=["/var/www",
          "/var/log/apache",
          "/dev/log"]
users=["chrapach"]
groups=["chrapach"]
packages=["apache", "apache-common"]
userFiles=["/etc/password",
           "/etc/shadow"]
groupFiles=["/etc/group",
            "/etc/gshadow"]
forceCopy=["/etc/hosts",
           "/etc/mime.types"]
```

  FIXME: algumas opções parecem não funcionar corretamente. Por exemplo, `/etc/shadow` e `/etc/gshadow` não são copiados, visto que `/etc/password` e `/etc/group` são copiados em vez de serem filtrados.

- Crie a árvore da jaula: `makejail /etc/makejail/apache.py`

- Se `/etc/password` e `/etc/group` forem copiados completamente, digite:

```
$ grep chrapach /etc/passwd > /var/chroot/apache/etc/passwd
$ grep chrapach /etc/group > /var/chroot/apache/etc/group
```

  para substituí-los com as cópias filtradas.

- Copie as páginas e os logs do site Web dentro da jaula. Estes arquivos não são copiados automaticamente (veja a opção *preserve* no arquivo de configuração do **makejail**).

```
# cp -Rp /var/www /var/chroot/apache/var
# cp -Rp /var/log/apache/*.log /var/chroot/apache/var/log/apache
```

- Edite o script de inicialização para que o daemon de logging do sistema também ouça do socket `/var/chroot/apache/dev/log`. No arquivo `/etc/init.d/sysklogd`, substitua: `SYSLOGD=""` com `SYSLOGD=" -a /var/chroot/apache/dev/log"` e reinicie o daemon (`/etc/init.d/sysklogd restart`).

- Edite o script de inicialização do Apache (`/etc/init.d/apache`). Você pode precisar fazer algumas alterações no script de inicialização padrão para que ele funcione apropriadamente com a árvore enjaulada. Como:

  - configure uma nova variável *CHRDIR* no início do arquivo;

  - edite as seções *start*, *stop*, *reload*, etc.;

  - adicione uma linha para montar e desmontar o sistema de arquivo `/proc` que está dentro da jaula.

```
#! /bin/bash
#
# apache\tStart the apache HTTP server.
#

CHRDIR=/var/chroot/apache

NAME=apache
PATH=/bin:/usr/bin:/sbin:/usr/sbin
DAEMON=/usr/sbin/apache
SUEXEC=/usr/lib/apache/suexec
PIDFILE=/var/run/$NAME.pid
CONF=/etc/apache/httpd.conf
APACHECTL=/usr/sbin/apachectl

trap "" 1
export LANG=C
export PATH

test -f $DAEMON || exit 0
test -f $APACHECTL || exit 0

# ensure we don't leak environment vars into apachectl
APACHECTL="env -i LANG=${LANG} PATH=${PATH} chroot $CHRDIR $APACHECTL"
```

```
if egrep -q -i "^[[:space:]]*ServerType[[:space:]]+inet" $CONF
then
    exit 0
fi

case "$1" in
  start)
    echo -n "Starting web server: $NAME"
    mount -t proc proc /var/chroot/apache/proc
    start-stop-daemon --start --pidfile $PIDFILE --exec $DAEMON \
\t--chroot $CHRDIR
    ;;

  stop)
    echo -n "Stopping web server: $NAME"
    start-stop-daemon --stop --pidfile "$CHRDIR/$PIDFILE" --oknodo
    umount /var/chroot/apache/proc
    ;;

  reload)
    echo -n "Reloading $NAME configuration"
    start-stop-daemon --stop --pidfile "$CHRDIR/$PIDFILE" \
\t--signal USR1 --startas $DAEMON --chroot $CHRDIR
    ;;

  reload-modules)
    echo -n "Reloading $NAME modules"
    start-stop-daemon --stop --pidfile "$CHRDIR/$PIDFILE" --oknodo \
\t--retry 30
    start-stop-daemon --start --pidfile $PIDFILE \
\t--exec $DAEMON --chroot $CHRDIR
    ;;

  restart)
    $0 reload-modules
    exit $?
    ;;

  force-reload)
    $0 reload-modules
    exit $?
    ;;

  *)
    echo "Usage: /etc/init.d/$NAME {start|stop|reload|reload-modules|force-reloa
    exit 1
    ;;
esac

if [ $? == 0 ]; then
\techo .
\texit 0
else
\techo failed
```

```
\texit 1
fi
```

FIXME: should the first Apache process be run as another user than root (i.e. add --chuid chrapach:chrapach)? Cons: chrapach will need write access to the logs, which is awkward.

- Substitua no `/etc/logrotate.d/apache` o `/var/log/apache/*.log` com `/var/chroot/apache/var/log/apache/*.log`

- Inicialize o Apache (**/etc/init.d/apache start**) e verifique o que está sendo reportado no log da jaula (`/var/chroot/apache/var/log/apache/error.log`). Se a sua configuração for mais complexa (exemplo: se também utiliza PHP e MySQL), alguns arquivos provavelmente estarão faltando. Se estes arquivos não são copiados automaticamente pelo **makejail**, você pode listá-los com a opção *forceCopy* (para copiar os arquivos diretamente) ou *packages* (para copiar pacotes completos e suas dependências) no arquivo de configuração `/etc/makejail/apache.py`.

- Digite `ps aux | grep apache` para ter certeza que o Apache está rodando. Você deve ver algo do tipo:

```
root 180 0.0 1.1 2936 1436 ? S 04:03 0:00 /usr/sbin/apache
chrapach 189 0.0 1.1 2960 1456 ? S 04:03 0:00 /usr/sbin/apache
chrapach 190 0.0 1.1 2960 1456 ? S 04:03 0:00 /usr/sbin/apache
chrapach 191 0.0 1.1 2960 1456 ? S 04:03 0:00 /usr/sbin/apache
chrapach 192 0.0 1.1 2960 1456 ? S 04:03 0:00 /usr/sbin/apache
chrapach 193 0.0 1.1 2960 1456 ? S 04:03 0:00 /usr/sbin/apache
```

- Certifique-se que os processos do Apache estão sendo executados na jaula chroot procurando no sistema de arquivo `/proc`: `ls -la /proc/process_number/root/.` onde *process_number* é um dos PID listados acima (por exemplo: segunda coluna; PID 189). As entradas para a árvore restrita devem ser listadas:

```
drwxr-sr-x 10 root staff 240 Dec 2 16:06 .
drwxrwsr-x 4 root staff 72 Dec 2 08:07 ..
drwxr-xr-x 2 root root 144 Dec 2 16:05 bin
drwxr-xr-x 2 root root 120 Dec 3 04:03 dev
drwxr-xr-x 5 root root 408 Dec 3 04:03 etc
drwxr-xr-x 2 root root 800 Dec 2 16:06 lib
dr-xr-xr-x 43 root root 0 Dec 3 05:03 proc
drwxr-xr-x 2 root root 48 Dec 2 16:06 sbin
drwxr-xr-x 6 root root 144 Dec 2 16:04 usr
drwxr-xr-x 7 root root 168 Dec 2 16:06 var
```

Para automatizar este teste, você pode digitar:`ls -la /proc/`cat /var/chroot/apache/var/run/apache.pid`/root/.`

FIXME: Add other tests that can be run to make sure the jail is closed?

A razão pela qual eu gosto disso é que a configuração da jaula não é tão complicada e o servidor pode ser atualizado em somente duas linhas:

```
apt-get update && apt-get install apache
makejail /etc/makejail/apache.py
```

Veja também

If you are looking for more information you can consider these sources of information in which the information presented is based: http://www.floc.net/makejail/, this program was written by Alain Tesio