Guide de sécurité du Debian

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

Guide de sécurité du Debian

par Javier Fernández-Sanguino Peña

Résumé

Ce document décrit la sécurité dans le projet Debian ainsi que dans le système d'exploitation Debian. Il commence par la sécurisation et le renforcement de l'installation standard d'une distribution Debian GNU/Linux. Il couvre quelques tâches courantes telles que la sécurisation d'un réseau utilisant Debian GNU/Linux et il donne également des informations complémentaires sur les outils de sécurisation disponibles ainsi que sur le travail accompli au sein du projet Debian par l'équipe en charge de la sécurité et par l'équipe d'audit.

Table des matières

Liste des exemples

Chapitre 1. Introduction

L'une des choses les plus difficiles dans l'écriture de documents liés à la sécurité est que chaque cas est unique. Il faut prêter attention à deux choses : la menace que constitue l'environnement et les besoins de sécurité liés à un site individuel, une machine ou un réseau. Par exemple, les exigences que l'on a pour une utilisation familiale n'ont rien de comparable aux exigences que l'on retrouve dans le réseau d'une banque. Alors que dans le premier cas, l'utilisateur aura à affronter de simples scripts d'attaque, le réseau d'une banque sera, lui, sous la menace d'attaques directes. De plus, la banque se doit de protéger l'exactitude des données de ses clients. Il faudra donc que chaque utilisateur trouve le bon compromis entre la facilité d'utilisation et la sécurité poussée à l'extrême.

Prenez conscience que cet ouvrage traite uniquement des questions liées aux logiciels. Le meilleur programme du monde ne pourra pas vous protéger contre quelqu'un qui aura un accès physique à la machine. Vous pouvez mettre la machine sous le bureau ou dans un bunker protégé par une armée. Pourtant, un ordinateur de bureau avec une bonne configuration sera beaucoup plus sûr (d'un point de vue logiciel) qu'un ordinateur protégé physiquement si son disque dur est truffé de logiciels connus pour avoir des failles de sécurité. Bien entendu, vous devez prendre en compte les deux aspects.

Ce document donne simplement un aperçu de ce qu'il est possible de faire pour accroître la sécurité du système Debian GNU/Linux. Si vous avez déjà lu des ouvrages traitant de la sécurité sous Linux, vous trouverez des similitudes avec ce document. Ce manuel ne prétend pas être l'ultime source d'informations à laquelle vous devez vous référer. Il essaye seulement d'adapter ces informations pour le système Debian GNU/Linux. D'autres distributions procèdent de manière différente pour certaines questions (le démarrage de démons est un exemple courant) ; vous trouverez dans cet ouvrage les éléments propres aux procédures et aux outils de Debian.

Auteurs

Le responsable actuel de ce document est mailto:jfs@debian.org. Veuillez lui envoyer vos commentaires, ajouts et suggestions et ils seront examinés pour une possible inclusion dans les prochaines versions de ce manuel.

Ce manuel a été lancé en tant que *HOWTO* par mailto:ar@rhwd.de. Après sa publication sur Internet, mailto:jfs@debian.org l'a incorporé dans le http://www.debian.org/doc. Un certain nombre de personnes ont contribué à ce manuel (la liste de toutes les contributions est dans le journal de modifications), mais les personnes suivantes méritent une mention spéciale car elles ont fourni des contributions significatives (des sections, chapitres ou annexes complets) :

- Stefano Canepa ;
- Era Eriksson ;
- Carlo Perassi ;
- Alexandre Ratti ;
- Jaime Robles ;
- Yotam Rubin ;
- Frederic Schutz ;
- Pedro Zorzenon Neto ;

- Oohara Yuuma ;
- Davor Ocelic.

Où récupérer ce manuel (et formats disponibles)

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.

Le document est également fourni aux formats texte, HTML et PDF dans le paquet http://packages.debian.org/harden-doc. Cependant, notez que le paquet peut ne pas être tout à fait à jour par rapport au document fourni sur le site Debian (mais vous pouvez toujours utiliser le paquet source pour construire vous-même une version à jour).

Ce document fait partie des documents distribués par le https://www.debian.org/doc/ddp. Les modifications introduites à ce document sont consultables à l'aide d'un navigateur web depuis les https://salsa.debian.org/ddp-team/securing-debian-manual. Vous pouvez aussi obtenir l'intégralité du code en utilisant Git :

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

Avis et réactions

Maintenant, la partie officielle. Pour l'instant, c'est Alexander Reelsen qui a écrit la plupart des paragraphes de ce manuel mais, selon lui, cela devrait évoluer. Il a grandi et vécu avec les logiciels libres : « c'est une part de ma vie quotidienne et, j'espère, de la vôtre aussi ». Il encourage chacun à lui envoyer ses réactions, astuces, ajouts ou suggestions.

Si vous pensez que vous pouvez vous occuper d'une partie en particulier ou d'un paragraphe, écrivez au responsable du document. Cela sera apprécié ! En particulier, si vous trouvez une section estampillée « FIXME », qui signifie que les auteurs n'ont pas eu le temps ou les connaissances requises pour s'en occuper, envoyez-leur un courrier immédiatement.

Le thème de ce manuel fait clairement comprendre qu'il est important de tenir ce manuel à jour ; vous pouvez apporter votre pierre à l'édifice. S'il vous plaît, aidez-nous.

Connaissances requises

L'installation de Debian GNU/Linux n'est pas très difficile et vous avez sans doute été capable de l'installer. Si vous disposez déjà de connaissances concernant Linux ou d'autres systèmes UNIX et si vous êtes

quelque peu familier avec les problèmes élémentaires de sécurité, il vous sera plus facile de comprendre ce manuel, car ce document ne peut pas entrer dans tous les petits détails (sans quoi cela aurait été un livre plutôt qu'un manuel). Si vous n'êtes pas si familier que cela avec ces systèmes, vous pouvez consulter la section intitulée « Connaissances requises » pour savoir où trouver des informations plus approfondies sur le sujet.

Éléments à écrire (FIXME/TODO)

Cette section décrit toutes les choses à corriger dans ce manuel. Certains paragraphes incluent des marques *FIXME* ou *TODO* décrivant quel contenu est manquant (ou quel type de travail doit être réalisé). Le but de cette section est de décrire toutes les choses qui devraient être incluses à l'avenir dans le manuel ou les améliorations à faire (ou qu'il serait intéressant d'ajouter).

Si vous pensez que vous pouvez apporter une contribution au contenu en corrigeant tout élément de cette liste (ou des annotations dans le texte lui-même), veuillez contacter l'auteur principal (la section intitulée « Auteurs »).

- Ce document doit encore être mis à jour en fonction des dernières publications de Debian. La configuration par défaut de certains paquets doit être adaptée car elles ont été modifiées depuis que ce document a été écrit.
- 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].
- Envisager la rédaction d'une section sur la construction d'applications orientées réseau pour Debian (avec des informations telles que le système de base, equivs et 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].
- Comment mettre en place un pare-feu en utilisant Debian GNU/Linux. La section sur les pare-feu concerne actuellement un système isolé (pas de protection d'autres machines, etc.). Comment tester la configuration.
- Paramétrage d'un serveur mandataire pare-feu avec Debian GNU/Linux et faire un état des lieux des paquets fournissant des services *proxy* (tels que xfwp, ftp-proxy, redir, smtpd, dnrd, jftpgw, oops, pdnsd, perdition, transproxy, tsocks). Renvoi au manuel pour toute autre information. Considérer également que zorp est maintenant disponible comme paquet Debian et qu'il *s'agit* d'un mandataire pare-feu (il existe également des paquets Debian fournis par les auteurs).
- Informations sur la configuration des services avec file-rc.
- Vérifier toutes les URLs et supprimer ou corriger celles qui ne sont plus disponibles.
- Ajouter des informations sur les substituts de serveurs typiques (disponibles dans Debian) qui fournissent des fonctionnalités restreintes. Par exemple :
  - lpr local par CUPS (paquet) ? ;

  - lrp distant par lpr ;
  - BIND par dnrd/maradns ;
  - Apache par dhttpd/thttpd/wn (tux?) ;
  - Exim/Sendmail par ssmtpd/smtpd/postfix ;
  - Squid par tinyproxy ;
  - ftpd par oftpd/vsftpd ;
  - etc.
- De plus amples informations concernant les correctifs spécialisés dans la sécurité du noyau dans Debian, incluant ceux montrés ci-dessus et ajouter des informations spécifiques sur la façon d'activer ces correctifs dans un système Debian.
  - Linux Intrusion Detection (kernel-patch-2.4-lids) ;
  - Linux Trustees (paquet trustees) ;
  - NSA Enhanced Linux [http://wiki.debian.org/SELinux]
  - linux-patch-openswan.
  - etc.
- Précisions sur l'arrêt de services réseaux inutiles (outre **inetd**) ; c'est en partie dans la procédure de consolidation mais pourrait être élargi un petit peu.
- Informations sur le renouvellement des mots de passe ; c'est étroitement lié à la politique mise en place.
- Politique de sécurité et formation des utilisateurs.
- Davantage à propos de tcpwrappers, et de l'encapsulation en général ?
- `hosts.equiv` et d'autres trous de sécurité majeurs.
- Problèmes avec les serveurs de partage de fichiers tels que Samba et NFS ?
- suidmanager/dpkg-statoverrides.
- lpr et lprng.
- Désactiver les outils GNOME qui utilisent 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).
- Parler des services « chrootés », plus d'informations sur http://www.linuxfocus.org/English/January2002/article225.shtml.
- Parler des programmes pour faire des « prisons » chroot. compartment et chrootuid sont en attente dans incoming. D'autres (makejail, jailer) pourraient aussi être présentés.

- Plus d'informations concernant les logiciels d'analyse de journaux (c'est-à-dire logcheck et logcolorise).
- Routage « avancé » (la politique de trafic concerne la sécurité).
- Restreindre **SSH** pour qu'il puisse uniquement exécuter certaines commandes.
- Utilisation de dpkg-statoverride.
- Moyens sûrs de partager un graveur de CD parmi les utilisateurs.
- Moyens sûrs de fournir du son en réseau en plus des possibilités d'affichage en réseau (pour que le son des clients X soit envoyé sur le périphérique de son du serveur X).
- Sécurisation des navigateurs web.
- Configurer FTP au travers de **SSH**.
- Utilisation des systèmes de fichiers « loopback » chiffrés.
- encrypting the entire file system.
- Outils stéganographiques.
- Configurer une autorité de clefs publiques (PKA) pour une organisation.
- Utiliser LDAP pour gérer les utilisateurs. Il y a un HOWTO sur ldap+kerberos pour Debian écrit par Turbo Fredrikson et disponible à http://www.bayour.com/LDAPv3-HOWTO.html.
- Comment enlever des informations non essentielles sur les systèmes de production tels que `/usr/share/doc`, `/usr/share/man` (oui, sécurité par obscurité).
- Plus d'informations sur lcap basées sur le fichier README des paquets (pas encore tout à fait présent, consultez le http://bugs.debian.org/cgi-bin/bugreport.cgi?bug=169465) et à partir de l'article de 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).
- Ajouter des informations sur l'exécution de plusieurs senseurs **snort** dans un système donné (vérifier les rapports de bogues envoyés à snort).
- Ajouter des informations sur la mise en place d'un pot de miel (« *honeypot* ») avec le paquet honeyd.
- Décrire la situation relative à FreeSwan (abandonné) et OpenSwan. La section VPN a besoin d'être récrite.
- Ajouter une section spécifique à propos des bases de données, l'installation par défaut et sur la façon de sécuriser les accès.
- Ajouter une section sur l'utilité des serveurs virtuels (Xen, Vserver, etc.).
- Expliquer comment utiliser plusieurs vérificateurs d'intégrité tels que aide, integrit ou samhain. La base est très simple à expliquer et permet de personnaliser la configuration par défaut.

Remerciements

- Alexander Reelsen qui a écrit le document original.

- a ajouté des informations au document original.
- Robert van der Meulen pour les paragraphes sur les quotas ainsi que de nombreuses bonnes idées.
- Ethan Benson qui a corrigé le paragraphe sur PAM et qui a eu quelques idées de qualité.
- Dariusz Puchalak qui a contribué aux informations de plusieurs chapitres.
- Gaby Schilders qui a eu une idée Géniale/Paranoïaque sympathique.
- Era Eriksson qui a éliminé les fautes de langage et qui a contribué à l'annexe « pense-bête ».
- Philipe Gaspar qui a écrit les informations concernant LKM.
- Yotam Rubin qui a contribué des correctifs pour de nombreuses fautes de frappe ainsi que les informations liées aux versions de BIND et aux mots de passe MD5.
- François Bayart pour l'annexe décrivant la mise en place d'un pont pare-feu.
- Joey Hess qui rédigea la section décrivant comment apt sécurisé fonctionne dans le http://wiki.debian.org/SecureApt.
- Martin F. Krafft qui ajouta quelques informations dans son blog à propos de la vérification des empreintes digitales (fingerprint) et qui furent réutilisées dans la section sur apt sécurisé.
- Francesco Poli qui fit une révision approfondie du manuel et fournit un grand nombre de rapports de bogue et de correctifs typographiques qui ont amélioré et aidé à mettre à jour le document.
- All the people who made suggestions for improvements that (eventually) were included here (see la section intitulée « Où récupérer ce manuel (et formats disponibles) »).
- Tous ceux qui m'ont encouragé (Alexander) à écrire ce HOWTO (qui devint plus tard ce manuel).
- Tout le projet Debian.

Chapitre 2. Avant de commencer

Que voulez-vous faire du système ?

Sécuriser un système Debian n'est pas différent de la sécurisation d'un autre système. Afin de procéder correctement, vous devez tout d'abord décider quelle en sera l'utilisation. Ensuite, vous devez penser aux tâches à prendre en compte si vous désirez réellement sécuriser le système.

Vous constaterez que ce manuel va du début à la fin, c'est-à-dire que vous trouverez des informations sur les tâches à réaliser avant, pendant et après l'installation du système Debian. Les tâches peuvent être découpées comme ceci :

- décider quels sont les services dont vous avez besoin et vous limiter à ceux-là. Cela comprend la désactivation ou la désinstallation des services inutiles et l'ajout de filtres de type pare-feu ou de tcpwrappers ;
- limiter les utilisateurs et les permissions sur le système ;
- consolider les services disponibles ; ainsi, même en cas d'intrusion, l'impact sur le système sera minimisé ;
- utiliser des outils appropriés pour garantir qu'une utilisation non autorisée sera détectée et que vous pourrez prendre des mesures adéquates.

Être conscient des problèmes de sécurité

Ce manuel n'explique pas pourquoi certains problèmes sont considérés comme des risques pour la sécurité. Toutefois, vous pourriez désirer avoir une meilleure vision de la sécurité sur les systèmes UNIX et plus particulièrement le système Linux. Prenez le temps de consulter les documentations relatives à la sécurité afin que, confronté à différents choix, vous puissiez prendre des décisions éclairées. Debian GNU/Linux est basée sur le noyau Linux ; aussi, la plupart des informations concernant Linux, venant d'autres distributions ou d'UNIX en général, peuvent être appliquées (même si les outils utilisés ou les programmes disponibles diffèrent).

Quelques documents pratiques.

- The http://www.tldp.org/HOWTO/Security-HOWTO/ is one of the best references regarding general Linux security.
- Le http://www.linuxsecurity.com/docs/LDP/Security-Quickstart-HOWTO/ est également une très bonne base pour les utilisateurs néophytes (aussi bien de Linux qu'en matière de sécurité).
- Le http://seifried.org/lasg/ est un guide complet qui englobe tous les problèmes de sécurité Linux, de la sécurité du noyau jusqu'aux VPN. Veuillez noter qu'il n'a pas été mis à jour depuis 2001, mais certaines informations peuvent encore être pertinentes. [1]
- l'article http://seifried.org/security/os/linux/20020324-securing-linux-step-by-step.html
- Dans http://www.tldp.org/links/p_books.html#securing_linux vous pouvez trouver un document similaire à ce manuel, mais destiné à Red Hat ; certaines questions ne sont pas spécifiques à cette distribution et peuvent s'appliquer à Debian.

[1] Il a été remplacé à un moment donné par le « Linux Security Knowledge Base ». Cette documentation est également disponible dans Debian par l'intermédiaire du paquet lskb. Il est à nouveau de retour en tant que le *Lasg*.

- 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/.

- Si vous voulez évaluer le programme que vous allez utiliser (ou en créer de nouveaux) vous devriez consulter le http://www.tldp.org/HOWTO/Secure-Programs-HOWTO.html (le document de référence est disponible à http://www.dwheeler.com/secure-programs/, il inclut des présentations et des conférences de l'auteur, David Wheeler).

- Si vous pensez installer un pare-feu, vous devriez consulter le http://www.tldp.org/HOWTO/Firewall-HOWTO.html et le http://www.tldp.org/HOWTO/IPCHAINS-HOWTO.html (pour les noyaux antérieurs au 2.4).

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

Dans tous les cas, vous trouverez plus d'informations concernant les services expliqués ici (NFS, NIS, SMB, etc.) dans les nombreux HOWTO du http://www.tldp.org/. Certains d'entre eux discutent de la sécurité d'un service donné, donc n'oubliez pas de jeter un œil là-dessus également.

Les documents HOWTO du Projet de documentation Linux sont disponibles dans Debian GNU/Linux avec l'installation des paquets doc-linux-text (version texte) ou doc-linux-html (version HTML). Après l'installation, ces documents seront respectivement disponibles dans les répertoires `/usr/share/doc/HOWTO/en-txt` et `/usr/share/doc/HOWTO/en-html`. De même, les versions françaises de ces documents sont disponibles dans les paquets doc-linux-fr-text (version texte) et doc-linux-fr-html (version HTML) qui seront respectivement disponibles dans les répertoires `/usr/share/doc/HOWTO/fr-txt` et `/usr/share/doc/HOWTO/fr-html`.

Autres livres Linux recommandés.

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

- Linux Security par John S. Flowers. New Riders ; ISBN : 0735700354. Mars 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

Livres divers (qui se rapportent à des questions générales concernant UNIX et la sécurité, non spécifiques à 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 ; 320 pages.

Quelques sites Internet utiles pour se tenir informé des questions de sécurité.

- 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.

Comment Debian gère la sécurité ?

Tout comme vous avez une vue générale de la sécurité dans Debian GNU/Linux, vous devez connaître les différents problèmes auxquels Debian s'attaque afin de fournir un système sécurisé.

- Les problèmes Debian sont toujours traités ouvertement, même ceux liés à la sécurité. Les problèmes de sécurité sont abordés ouvertement sur la liste de discussions debian-security. Les bulletins de sécurité Debian (DSA - Debian Security Advisories) sont envoyés sur des listes de discussions publiques (internes et externes) et publiés sur des serveurs publics. Tel que déclaré dans le http://www.debian.org/social_contract
- Debian suit les problèmes de sécurité de très près. L'équipe en charge de la sécurité consulte les sources relatives à la sécurité, la plus importante étant http://www.securityfocus.com/cgi-bin/vulns.pl, à la recherche de paquets possédant des problèmes de sécurité et qui pourraient être inclus dans Debian.
- Les mises à jour liées à la sécurité sont la première priorité. Lorsqu'un problème survient dans un paquet Debian, la mise à jour est réalisée aussi vite que possible et elle est intégrée dans nos versions *stable*, *testing* et *unstable* pour toutes les architectures.
- Les informations concernant la sécurité sont centralisées en un point unique, http://security.debian.org/.
- Debian essaie toujours d'améliorer la sécurité globale de la distribution en lançant de nouveaux projets, comme les vérifications automatiques des signatures de paquets.
- Debian fournit de nombreux outils liés à la sécurité pour l'administration système et la surveillance. Les développeurs essayent de lier étroitement ces outils à la distribution de façon à créer un ensemble améliorant les règles locales de sécurité. Les outils disponibles sont : vérificateurs d'intégrité, outils d'audit, outils de consolidation, outils pour pare-feu, outils de détection d'intrusion, etc.
- Les responsables de paquets sont avertis des problèmes de sécurité. Cela conduit à de nombreuses installations de service « sécurisé par défaut » ; cela peut parfois imposer certaines restrictions à une utilisation normale. Toutefois, Debian essaie d'équilibrer les problèmes de sécurité et la facilité d'administration : par exemple, les programmes ne sont pas installés en mode *désactivé* (comme c'est le cas avec la famille des systèmes d'exploitation BSD). Dans tous les cas, quelques problèmes spéciaux, tels les programmes *setuid*, sont abordés par la http://www.debian.org/doc/debian-policy/.

En publiant des informations de sécurité spécifiques à Debian et en complétant d'autres documents d'informations sur la sécurité relatifs à Debian (consultez la section intitulée « Connaissances requises »), ce document a pour but de favoriser des installations de systèmes beaucoup mieux sécurisées.

Chapitre 3. Avant et pendant l'installation

Choisir un mot de passe pour le BIOS

Avant d'installer un système d'exploitation sur l'ordinateur, créez un mot de passe pour le BIOS. Après l'installation (une fois que vous avez rendu possible un démarrage à partir du disque dur), retournez dans le BIOS et changez la séquence de démarrage afin de rendre impossible le démarrage à partir d'une disquette, d'un CD ou de tout autre périphérique. Sinon un pirate n'a besoin que d'un accès physique et d'une disquette de démarrage pour accéder au système complet.

Désactiver le démarrage sans mot de passe est une solution encore meilleure. Cela peut être très efficace pour un serveur car il n'est pas redémarré très souvent. L'inconvénient de cette méthode est que le redémarrage nécessite l'intervention d'une personne, ce qui peut poser des problèmes si la machine n'est pas facilement accessible.

Remarque : certains BIOS ont des mots de passe par défaut bien connus et des applications existent également pour récupérer les mots de passe du BIOS. Corollaire : ne dépendez pas de cette mesure pour sécuriser l'accès console du système.

Partitionner le système

Choisir un schéma de partitionnement intelligent

Un schéma de partitionnement intelligent dépend de l'utilisation de la machine. Une bonne règle est d'être assez large avec vos partitions et de faire attention aux facteurs suivants.

- Les arborescences de répertoires modifiables par un utilisateur, telles que `/home`, `/tmp` et `/var/tmp`, doivent être sur des partitions distinctes. Cela réduit le risque qu'un déni de service provoqué par un utilisateur ne remplisse le point de montage « / » rendant ainsi le système inutilisable (remarque : ce n'est pas strictement vrai car il existe toujours un espace réservé au superutilisateur qu'un utilisateur normal ne pourra pas remplir) et cela empêche les attaques de liens directs (*hardlinks*). [1]

- Toute partition qui peut fluctuer, par exemple `/var` (surtout `/var/log`) devrait être également sur une partition distincte. Sur un système Debian, vous devriez créer `/var` un petit peu plus grand que la normale car les paquets téléchargés (le cache d'apt) sont stockés dans `/var/cache/apt/archives`.

- Toute partition où vous voulez installer des logiciels ne faisant pas partie de la distribution devrait être sur une partition distincte. Selon la norme de hiérarchie des fichiers (FHS), c'est `/opt` ou `/usr/local`. Si ce sont des partitions distinctes, elles ne seront pas effacées si vous devez réinstaller Debian.

- D'un point de vue sécurité, il est souhaitable de mettre les données statiques sur une partition et de monter celle-ci en lecture seule. Encore mieux, mettre les données sur un périphérique en lecture seule. Voir ci-dessous pour plus d'informations.

[1] Un très bon exemple de ce type d'attaque utilisant /tmp est détaillé dans http://www.hackinglinuxexposed.com/articles/20031111.html et http://www.hackinglinuxexposed.com/articles/20031214.html (notez que l'incident est lié à Debian). C'est de manière basique une attaque dans laquelle un utilisateur local *cache* profondément une application setuid vulnérable en faisant un lien direct sur celle-ci, évitant de manière efficace toute mise à jour (ou suppression) du binaire lui-même réalisé par l'administrateur du système. dpkg a été récemment corrigé pour empêcher cela (consultez le http://bugs.debian.org/225692), mais d'autres binaires setuid (non contrôlés par le gestionnaire de paquets) sont risqués si les partitions ne sont pas mises en place correctement.

Dans le cas d'un serveur de courriers, il est important d'avoir une partition séparée pour le répertoire des courriers (spool). Les utilisateurs distants (soit consciemment, soit inconsciemment) peuvent remplir le répertoire des courriers (`/var/mail` ou `/var/spool/mail`). Si le répertoire est sur une partition séparée, cette situation ne rendra pas le système inutilisable. Sinon (si le répertoire est sur la même partition que `/var`), le système pourrait avoir d'importants problèmes : les entrées des journaux ne seront pas créées, aucun paquet ne pourra plus être installé et certains programmes pourraient même avoir des problèmes à être exécutés (s'ils utilisent `/var/run`).

Pour les partitions pour lesquelles vous ne pouvez pas être certain de la place nécessaire, installez Logical Volume Manager (lvm-common et les binaires nécessaires pour le noyau qui peuvent être lvm10, lvm6 ou lvm5). En utilisant `lvm`, vous pouvez créer des groupes de volumes répartis sur plusieurs volumes physiques.

Choisir les systèmes de fichiers appropriés

Pendant le partitionnement du système, vous devez également décider du système de fichiers à utiliser. Le système de fichiers choisi par défaut[2] pendant l'installation de Debian pour les partitions Linux est `ext3`, un système de fichiers journalisé. Vous devriez toujours utiliser un système de fichiers journalisé comme `ext3`, `reiserfs`, `jfs` ou `xfs` pour réduire les problèmes découlant d'un plantage système dans les cas suivants :

- pour les portables pour tous les systèmes de fichiers installés. Ainsi, si la batterie se vide inopinément ou si le système se gèle à cause d'un problème matériel (comme pour la configuration de X, ce qui est assez courant), vous êtes moins susceptible de perdre des données pendant le redémarrage matériel.

- pour les systèmes de production qui stockent des quantités importantes de données (comme les serveurs de courriers, les serveurs FTP, les systèmes de fichiers en réseau, etc.), cela est recommandé pour ces partitions. Ainsi, en cas de plantage du système, le serveur nécessitera moins de temps pour récupérer et vérifier le système de fichiers et une perte de données est moins probable.

En laissant de côté les problèmes de performance concernant les systèmes de fichiers journalisés (cela pouvant parfois tourner à la guerre de religion), il est habituellement préférable d'utiliser le système de fichiers `ext3`. La raison pour cela est qu'il est rétrocompatible avec `ext2`, donc s'il y a un quelconque problème avec la journalisation, vous pouvez la désactiver et toujours avoir un système de fichiers fonctionnel. De plus, si vous avez besoin de récupérer le système avec une disquette d'amorçage (ou un CD), vous n'avez pas besoin d'un noyau personnalisé. Si le noyau est en version 2.4 ou 2.6, la prise en charge `ext3` est déjà disponible, s'il s'agit d'un noyau 2.2, vous pourrez amorcer le système de fichiers même si vous n'aurez plus la capacité de journalisation. Si vous utilisez d'autres systèmes de fichiers, vous trouverez que vous ne pourrez pas effectuer de récupération à moins d'avoir un noyau 2.4 ou 2.6 avec les modules nécessaires inclus dans le noyau. Si vous êtes bloqué avec un noyau 2.2 sur la disquette de sauvegarde, cela pourrait même être encore plus difficile d'accéder à des partitions `reiserfs` ou `xfs`.

Dans tous les cas, il est possible que l'intégrité des données soit meilleure avec `ext3` car il fait de la journalisation des données par fichier alors que les autres ne font que de la journalisation par métadonnées, consultez http://lwn.net/2001/0802/a/ext3-modes.php3.

Remarquez, néanmoins, que certaines partitions n'ont pas d'intérêt particulier à utiliser un système de fichiers journalisé. Par exemple, si vous utilisez une partition à part pour `/tmp/`, vous devriez plutôt utiliser un système de fichiers `ext2` car elle sera nettoyée lors du démarrage du système.

[2] Depuis Debian GNU/Linux 4.0, surnommée `Etch`.

Ne pas se connecter à Internet tant que tout n'est pas prêt

Le système ne devrait pas être connecté à Internet pendant l'installation. Cela peut paraître stupide mais il faut savoir que l'installation par le réseau est une méthode d'installation habituelle. Étant donné que le système va installer et activer les services immédiatement, si le système est connecté à Internet et que les services ne sont pas configurés correctement, vous les exposez à des attaques.

Il faut noter également que certains services peuvent avoir des trous de sécurité qui n'ont pas encore été corrigés dans les paquets que vous utilisez pour l'installation. C'est généralement vrai si vous installez depuis de vieux supports (comme des CD). Dans ce cas, il se peut que le système soit compromis avant même la fin de l'installation !

Étant donné que l'installation et les mises à jour peuvent être faites par Internet, vous pourriez penser que c'est une bonne idée d'utiliser cette caractéristique lors de l'installation. Si le système va être connecté directement à Internet (et pas protégé par un pare-feu ou un NAT), il est plus judicieux de l'installer sans connexion à Internet et d'utiliser un miroir local de paquets contenant à la fois les paquets source et les mises à jour de sécurité. Vous pouvez mettre en place des miroirs de paquets en utilisant un autre système connecté à Internet et des outils spécifiques à Debian (si c'est un système Debian) tels que apt-move ou apt-proxy ou tout autre outil de miroir pour fournir l'archive aux systèmes installés. Si vous ne pouvez pas faire cela, vous pouvez mettre en place des règles de pare-feu pour limiter l'accès au système pendant la mise à jour (consultez la section intitulée « Mise à jour de sécurité protégée par un pare-feu »).

Définir un mot de passe pour le superutilisateur

Définir un bon mot de passe est la condition de base pour avoir un système sécurisé. Consultez passwd(1) pour quelques conseils pour créer de bons mots de passe. Vous pouvez également utiliser un générateur automatique de mots de passe pour faire cela pour vous (consultez la section intitulée « Générer des mots de passe utilisateur »).

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

Administrer le nombre minimal de services nécessaires

Les services sont des programmes tels que les serveurs FTP et les serveurs web. Puisqu'ils doivent *écouter* les connexions entrantes qui demandent le service, des ordinateurs externes peuvent se connecter au vôtre. Les services sont parfois vulnérables (entendez par là qu'ils peuvent être compromis par certaines attaques) : ils créent des risques pour la sécurité.

Vous ne devriez pas installer les services dont vous n'avez pas besoin sur la machine. Chaque service installé peut introduire de nouveaux trous de sécurité, peu évidents ou inconnus, sur l'ordinateur.

Comme vous le savez sans doute déjà, lorsque vous installez un service, le comportement par défaut est de l'activer. Dans une installation Debian par défaut, sans service installé, le nombre de services actifs est assez faible et il est même plus faible quand on parle de services réseau. Dans une installation stan-

dard de Debian 3.1, les seuls services activés par défaut sont OpenSSH, Exim (selon la façon dont vous l'avez configuré) et le portmapper RPC comme services réseau[3]. Si vous n'avez pas choisi l'installation standard, mais que vous avez sélectionné l'installation en mode expert, vous obtiendrez une installation avec aucun service réseau actif. Le portmapper RPC est installé par défaut car il est nécessaire pour beaucoup de services, par exemple NFS. Cependant, il peut facilement être retiré, consultez la section intitulée « Sécurisation des services RPC » pour plus d'informations sur la façon de sécuriser ou de désactiver les services RPC.

Lorsque vous installez un nouveau service réseau (démon) sur le système Debian GNU/Linux, il peut être activé de deux façons : avec le superdémon inetd (une ligne sera ajoutée à `/etc/inetd.conf`) ou par un programme qui s'attache lui-même aux interfaces réseau. Ces programmes sont contrôlés par les fichiers `/etc/init.d` qui sont appelés lors du démarrage au moyen du mécanisme System V (ou un autre) en utilisant des liens symboliques dans `/etc/rc?.d/*` (pour plus d'informations sur la manière dont cela est fait, consultez `/usr/share/doc/sysvinit/README.runlevels.gz`).

Si vous voulez garder certains services tout en les utilisant rarement, utilisez les commandes **update-***, par exemple **update-inetd** et **update-rc.d** pour les supprimer du processus de démarrage. Pour plus d'informations sur la façon de désactiver des services réseau, veuillez consulter la section intitulée « Désactivation de services démon ». Si vous voulez changer le comportement par défaut de démarrage des services à l'installation de leur paquet associé[4], utilisez **policy-rc.d**, veuillez consulter `/usr/share/doc/sysv-rc/README.policy-rc.d.gz` pour obtenir plus de renseignements.

La prise en charge d'**invoke-rc.d** est obligatoire dans Debian, ce qui veut dire que pour Debian 4.0 *Etch* et versions suivantes, vous pouvez écrire un fichier policy-rc.d qui interdit le démarrage des nouveaux démons avant de les avoir configurés. Même si aucun de ces scripts n'est encore empaqueté, ils sont plutôt faciles à écrire. Consultez policyrcd-script-zg2.

Désactivation de services démon

La désactivation d'un service démon est relativement simple. Vous pouvez soit supprimer le paquet fournissant le programme pour ce service, soit supprimer ou renommer les liens de démarrage sous `/etc/rc${runlevel}.d/`. Si vous les renommez, assurez-vous qu'ils ne commencent pas avec un « S » pour qu'ils ne soient pas démarrés par **/etc/init.d/rc**. Ne supprimez pas tous les liens disponibles ou le système de gestion des paquets les régénérera lors des mises à jour du paquet, assurez-vous de laisser au moins un lien (typiquement, un lien « K », « kill » pour tuer). Pour obtenir plus de renseignements, veuillez consulter la section http://www.debian.org/doc/manuals/reference/ch-system#s-custombootscripts de la référence Debian (chapitre 2 - fondamentaux de Debian).

Vous pouvez supprimer ces liens manuellement ou en utilisant `update-rc.d` (consultez update-rc.d(8)). Vous pouvez, par exemple, désactiver un service pour les niveaux d'exécution multiutilisateurs en faisant :

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

Avec *XX* un nombre qui détermine quand l'action d'arrêt pour ce service sera exécutée. Veuillez noter que, si vous *n'utilisez pas* file-rc, `update-rc.d -f service remove` ne fonctionnera pas correctement car *tous* les liens sont supprimés, lors d'une réinstallation ou d'une mise à jour du paquet, ces liens seront régénérés (ce qui n'est probablement pas ce que vous voulez). Si vous pensez que cela n'est pas intuitif, vous avez probablement raison (consultez le http://bugs.debian.org/67095). D'après les pages de manuel :

[3] L'empreinte dans Debian 3.0 et les versions précédentes n'était pas aussi réduite car certains services **inetd** étaient activés par défaut. Les installations standard de Debian 2.2 installaient également le serveur NFS ainsi que le serveur TELNET.
[4] C'est pratique si vous mettez en place un chroot de développement, par exemple.

```
Si des fichiers /etc/rcrunlevel.d/[SK]??nom existent déjà,
alors update-rc.d ne fait rien. C'est ainsi fait pour que
l'administrateur système puisse réarranger les liens — à condition
qu'il en reste au moins un — sans que sa configuration ne soit
réécrite.
```

Si vous utilisez file-rc, toutes les informations concernant le démarrage des services sont gérées par un fichier de configuration commun et sont conservées même si les paquets sont retirés du système.

Vous pouvez utiliser l'interface texte (TUI, Text User Interface) fournie par sysv-rc-conf pour faire tous ces changements facilement (**sysv-rc-conf** fonctionne pour file-rc ainsi que pour les niveaux d'exécution normaux de type System V). Vous pouvez également trouver des interfaces graphiques similaires pour les systèmes de bureau. Vous pouvez aussi utiliser l'interface en ligne de commande de sysv-rc-conf :

```
# sysv-rc-conf bidule off
```

L'avantage, avec cet utilitaire, est que les liens rc.d sont remis dans l'état qu'ils avaient avant l'appel « off » si vous réactivez le service avec :

```
# sysv-rc-conf bidule on
```

D'autres méthodes (moins recommandées) pour désactiver les services sont :

- suppression du script `/etc/init.d/nom_service` et suppression des liens de démarrage avec :

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

- déplacement du fichier script (`/etc/init.d/nom_service`) vers un autre nom (par exemple `/etc/init.d/OFF.nom_service`). Cela laissera des liens symboliques non valables sous `/etc/rc${runlevel}.d/` et générera des messages d'erreur au démarrage du système ;

- suppression du droit d'exécution du fichier `/etc/init.d/nom_service`. Cela générera également des messages d'erreur au démarrage ;

- édition du script `/etc/init.d/nom_service` pour qu'il s'arrête immédiatement lorsqu'il est exécuté (en ajoutant une ligne **exit 0** au début ou en commentant la partie `start-stop-daemon` dans celui-ci). Si vous procédez de cette façon, vous ne pourrez plus utiliser le script pour démarrer le service vous-même plus tard.

Cependant, les fichiers sous `/etc/init.d` sont des fichiers de configuration et ne devraient pas être écrasés lors des mises à jour de paquet si vous y avez fait des modifications locales.

Contrairement à d'autres systèmes d'exploitation (UNIX), les services dans Debian ne peuvent pas être désactivés en modifiant les fichiers dans `/etc/default/nom_service`.

FIXME : Ajouter des informations sur la gestion des démons par file-rc.

Désactivation d'inetd ou de ses services

Vous devriez vérifier si vous avez vraiment besoin du démon **inetd** de nos jours. **inetd** a toujours été un moyen de compenser des déficiences du noyau, mais celles-ci ont été corrigées dans les noyaux Linux modernes. Des possibilités de déni de service existent avec **inetd** (qui peut augmenter énormément la charge de la machine) et de nombreuses personnes préfèrent utiliser des démons indépendants au lieu d'appeler

des services avec **inetd**. Si vous voulez toujours exécuter un service du genre d'**inetd**, tournez-vous plutôt vers un démon inetd plus configurable comme **xinetd**, **rlinetd** ou **openbsd-inetd**.

Vous devriez arrêter tous les services inetd non nécessaires sur le système, comme **echo**, **chargen**, **discard**, **daytime**, **time**, **talk**, **ntalk** et les r-services (services à distance) (**rsh**, **rlogin** et **rcp**) qui sont considérés comme EXTRÊMEMENT dangereux (utilisez **ssh** à la place).

Vous pouvez désactiver les services en modifiant directement `/etc/inetd.conf`, mais Debian offre un meilleur moyen : `update-inetd` (qui commente les services de manière à ce qu'ils puissent être facilement réactivés). Vous pouvez supprimer le démon **telnet** en exécutant cette commande pour changer le fichier de configuration et redémarrer le démon (dans ce cas le service est désactivé) :

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

Si vous désirez des services en attente, mais qui n'écoutent pas sur toutes les adresses IP de l'hôte, vous voudrez peut-être utiliser des fonctions non documentées de **inetd** (remplacez des noms de service avec la syntaxe service@ip) ou utilisez un autre démon tel que **xinetd**.

Installer le minimum de logiciels nécessaires

Debian est fournie avec une *grande quantité* de logiciels, par exemple, Debian 3.0 *Woody* inclut 6 ou 7 (selon les architectures) CD de logiciels et des milliers de paquets et la version 3.1 fournit environ 13 CD de logiciels. Avec autant de logiciels et même si l'installation du système de base est assez réduite [5] vous pourriez vous laisser entraîner et installer plus de logiciels qu'il n'est vraiment nécessaire sur le système.

Comme vous connaissez déjà l'utilisation du système (n'est-ce pas ?), vous ne devez installer que les logiciels qui sont vraiment nécessaires à son fonctionnement. Tout outil non nécessaire installé pourrait être utilisé par un utilisateur qui voudrait compromettre le système ou par un intrus externe qui aurait obtenu un accès à l'interpréteur de commandes (ou par exécution de code à distance grâce à un service exploitable).

La présence, par exemple, d'outils de développement (un compilateur C) ou de langages interprétés (comme **perl** – voir ci-dessous – **python**, **tcl**, etc.) pourrait aider un attaquant à compromettre le système un peu plus :

- lui permettre d'augmenter ses droits. Il est plus facile, par exemple, d'exploiter localement le système si un débogueur et un compilateur sont prêts à les compiler et à les tester !
- fournir des outils qui pourraient aider l'attaquant à utiliser le système compromis comme une *base d'attaque* contre d'autres systèmes.[6]

Bien sûr, un intrus ayant un accès local à l'interpréteur de commandes peut télécharger son propre jeu d'outils et les exécuter, et l'interpréteur de commandes peut lui-même être utilisé pour créer des programmes complexes. Supprimer les logiciels non nécessaires ne va pas aider à *prévenir* le problème, mais cela rendra la tâche un peu plus difficile pour un attaquant (et certains pourraient abandonner dans cette situation et aller chercher des cibles plus faciles). Ainsi, si vous laissez des outils sur un système de production qui peuvent être utilisés pour attaquer des systèmes à distance (consultez la section intitulée « Outils d'évaluation des vulnérabilités à distance »), vous pouvez vous attendre à ce qu'un intrus les utilise également s'ils sont disponibles.

[5] Par exemple, dans Debian *Woody*, elle est d'environ 400 à 500 Mo, essayez ceci :

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

[6] Beaucoup d'intrusions ne sont faites que pour avoir accès aux ressources pour effectuer des activités illégales (attaques de déni de service, envoi d'indésirables, serveurs FTP illicites, pollution de DNS, etc.) plus que pour obtenir des données confidentielles du système compromis.

Veuillez noter qu'une installation par défaut de Debian *Sarge* (c'est-à-dire une installation pour laquelle aucun paquet individuel n'est sélectionné) installera un certain nombre d'outils de développement qui ne sont habituellement pas nécessaires. Cela vient du fait que certains paquets de développement sont de priorité *Standard*. Si vous ne comptez pas faire de développement, vous pouvez supprimer ces paquets du système sans inquiétude, ce qui devrait également aider à libérer de la place :

```
Paquet                    Taille
------------------------+--------
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 : paquet virtuel)
linux-kernel-headers    1,377,022
bin86                      82,090
cpp                        29,446
gcc                         4,896 (Note : paquet virtuel)
g++-3.3                 1,778,880
bison                     702,830
make                      366,138
libstdc++5-3.3-dev        774,982
```

Ce problème est corrigé dans les versions après *Sarge*, consultez le http://bugs.debian.org/cgi-bin/bugreport.cgi?bug=301273 et le http://bugs.debian.org/cgi-bin/bugreport.cgi?bug=301138. À cause d'un bogue dans le système d'installation, cela ne se produisait pas lors de l'installation de Debian 3.0 *Woody*.

Supprimer Perl

Vous devez prendre en compte qu'enlever **perl** peut ne pas être très simple (en fait, cela peut être assez difficile) sur un système Debian car il est utilisé par beaucoup d'outils système. Le paquet perl-base est également *Priority: required* (ce qui veut tout dire). C'est tout de même faisable, mais vous ne pourrez pas exécuter d'applications **perl** sur le système ; vous devrez également tromper le système de gestion des paquets pour lui faire croire que le paquet perl-base est installé même si ce n'est pas le cas. [7]

Quels outils utilisent **perl** ? Vous pouvez vous en rendre compte vous-même :

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

Ceux-ci incluent les outils suivants des paquets de priorité *requis* ou *important* :

- `/usr/bin/chkdupexe` du paquet util-linux.
- `/usr/bin/replay` du paquet bsdutils.
- `/usr/sbin/cleanup-info` du paquet dpkg.
- `/usr/sbin/dpkg-divert` du paquet dpkg.

[7] Vous pouvez créer (sur un autre système) un paquet bidon avec equivs.

- `/usr/sbin/dpkg-statoverride` du paquet dpkg.
- `/usr/sbin/install-info` du paquet dpkg.
- `/usr/sbin/update-alternatives` du paquet dpkg.
- `/usr/sbin/update-rc.d` du paquet sysvinit.
- `/usr/bin/grog` du paquet groff-base.
- `/usr/sbin/adduser` du paquet adduser.
- `/usr/sbin/debconf-show` du paquet debconf.
- `/usr/sbin/deluser` du paquet adduser.
- `/usr/sbin/dpkg-preconfigure` du paquet debconf.
- `/usr/sbin/dpkg-reconfigure` du paquet debconf.
- `/usr/sbin/exigrep` du paquet exim.
- `/usr/sbin/eximconfig` du paquet exim.
- `/usr/sbin/eximstats` du paquet exim.
- `/usr/sbin/exim-upgrade-to-r3` du paquet exim.
- `/usr/sbin/exiqsumm` du paquet exim.
- `/usr/sbin/keytab-lilo` du paquet lilo.
- `/usr/sbin/liloconfig` du paquet lilo.
- `/usr/sbin/lilo_find_mbr` du paquet lilo.
- `/usr/sbin/syslogd-listfiles` du paquet sysklogd.
- `/usr/sbin/syslog-facility` du paquet sysklogd.
- `/usr/sbin/update-inetd` du paquet netbase.

Donc, sans Perl et à moins que vous ne réécriviez ces outils en script shell, vous ne pourrez probablement pas gérer de paquets (vous ne pourrez donc pas mettre à jour le système, ce qui n'est *pas une Bonne Chose*).

Si vous êtes déterminé à enlever Perl du système de base Debian et si vous avez du temps libre, créez des rapports de bogue sur les paquets précédents en incluant un remplacement (sous forme de correctif) écrit en script shell aux outils ci-dessus.

Si vous désirez vérifier quels paquets Debian dépendent de Perl, vous pouvez utiliser :

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

ou

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

Consulter les listes de discussions Debian sur la sécurité

Cela ne fait pas de mal de jeter un œil à la liste de discussion debian-security-announce, où des alertes et des solutions pour les paquets sont annoncés par l'équipe sécurité de Debian, ou sur la liste mailto:debian-security@lists.debian.org, où vous pouvez participer aux discussions à propos de différentes choses liées à la sécurité Debian.

De façon à recevoir les alertes importantes concernant les mises à jour liées à la sécurité, envoyez un courriel à mailto:debian-security-announce-request@lists.debian.org avec le mot « subscribe » dans le sujet du courrier. Vous pouvez également vous inscrire à cette liste sur la page web sur http://www.debian.org/MailingLists/subscribe.

Cette liste de discussion a très peu de trafic, et en vous inscrivant vous serez tenu au courant des mises à jour pour la distribution Debian. Cela vous permet de télécharger rapidement les nouveaux paquets avec correction des bogues de sécurité, ce qui est relativement important dans le maintien d'un système sécurisé (consultez la section intitulée « Faire une mise à jour de sécurité » pour obtenir plus de précisions).

Chapitre 4. Après l'installation

Une fois que le système est installé, vous pouvez encore en faire plus pour sécuriser le système; certaines des étapes décrites ci-dessous peuvent être effectuées. Bien sûr, cela dépend vraiment de la configuration, mais pour prévenir un accès physique, vous devriez consulter la section intitulée « Changer le BIOS (à nouveau) », la section intitulée « Attribuer un mot de passe à LILO ou GRUB », la section intitulée « Enlever l'invite superutilisateur du noyau », la section intitulée « Restreindre les accès aux consoles » et la section intitulée « Restreindre les redémarrages système depuis la console ».

Avant de vous connecter à tout réseau, particulièrement s'il s'agit d'un réseau public, vous devriez, au minimum, faire une mise à jour de sécurité (consultez la section intitulée « Faire une mise à jour de sécurité »). Vous pourriez facultativement faire un instantané du système (consultez la section intitulée « Prendre un instantané («snapshot») du système »).

S'abonner à la liste de diffusion Debian Security Announce

Pour recevoir des informations sur les mises à jour de sécurité disponibles, vous devriez vous abonner à la liste de diffusion debian-security-announce pour recevoir les bulletins de sécurité de Debian[1]. Consultez la section intitulée « L'équipe de sécurité Debian » pour plus d'informations sur le fonctionnement de l'équipe en charge de la sécurité Debian. Pour des informations sur l'inscription aux listes de diffusion Debian, consultez http://lists.debian.org.

Les DSA sont signées avec la clef de l'équipe de sécurité Debian qui peut être récupérée sur http://security.debian.org.

Vous devriez également envisager de vous abonner à la http://lists.debian.org/debian-security pour des discussions générales sur les problèmes de sécurité dans le système d'exploitation Debian. Vous pourrez entrer en contact avec d'autres administrateurs système ainsi qu'avec des développeurs Debian et des développeurs amont d'outils de sécurité qui pourront répondre à vos questions et proposer leurs conseils.

FIXME : Ajouter la clef ici également?

Faire une mise à jour de sécurité

Dès que de nouveaux bogues de sécurité sont décelés dans les paquets, les responsables Debian et les auteurs amont les corrigent généralement dans les journées ou les heures suivantes. Une fois le bogue résolu, un nouveau paquet est fourni sur http://security.debian.org.

Si vous installez une version de Debian, vous devez prendre en compte le fait qu'il a pu y avoir des mises à jour de sécurité depuis la parution, à chaque fois qu'une vulnérabilité a été découverte dans un paquet. Ainsi, des révisions mineures (il y en a eu quatre dans la version Debian 3.0 *Sarge*) incluent ces mises à jour de paquets.

Pendant l'installation, les mises à jour de sécurité sont configurées sur le système, et les mises à jour en attente sont téléchargées et appliquées, sauf indication contraire ou si le système n'est pas connecté à Internet. Les mises à jour sont appliquées avant même le premier démarrage, de telle sorte que le nouveau système commence sa vie aussi à jour que possible.

[1] Debian Security Advisories (DSA).

Pour mettre à jour vous-même le système, ajoutez la ligne suivante dans le `sources.list` et vous recevrez les mises à jour de sécurité automatiquement quand vous mettrez à jour le système. Remplacez *[NOM]* par le nom de la version, par exemple *squeeze*.

```
deb http://security.debian.org/ [NOM]/updates main contrib non-free
```

Remarque : si vous utilisez la distribution *testing*, utilisez les source du miroir de sécurité de testing conformément à la section intitulée « Suivi en sécurité de la branche testing ».

Après avoir fait cela, plusieurs outils vous permettent de mettre à niveau le système. S'il s'agit d'un ordinateur de bureau, une application appelée **update-notifier**[2] permet de vérifier facilement si de nouvelles mises à niveau sont disponibles. En choisissant cela, vous pouvez faire les mises à niveau depuis le bureau (en utilisant **update-manager**). Pour obtenir plus de renseignements, veuillez consulter la section intitulée « Vérification de mises à jour sur station de travail ». Dans les environnements de bureau, vous pouvez aussi utiliser synaptic (GNOME), kpackage ou adept (KDE) pour des interfaces plus avancées. Si le système ne possède qu'un terminal texte, vous pouvez utiliser aptitude, apt ou dselect (obsolète) pour mettre à niveau.

- Si vous voulez utiliser l'interface texte d'aptitude, il suffit d'appuyer sur *u* (mise à jour) suivi de *g* (pour mettre à niveau). Vous pouvez aussi utiliser simplement la ligne de commande (en tant que superutilisateur) :

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

- Si vous voulez utiliser apt, il suffit de faire comme pour **aptitude**, mais en remplaçant **aptitude** des lignes précédentes par **apt-get**.

- Si vous voulez utiliser dselect, choisissez tout d'abord mise à jo[U]r, puis [I]nstaller et enfin [C]onfigurer pour mettre à jour et installer les paquets.

Si vous le voulez, vous pouvez ajouter également les lignes deb-src à `/etc/apt/sources.list`. Consultez apt(8) pour plus de détails.

Mise à jour de sécurité des bibliothèques

Une fois que vous avez exécuté une mise à jour de sécurité, il se peut que vous deviez redémarrer certains des services système. Si vous ne faites pas cela, certains services pourraient encore être vulnérables après une mise à jour de sécurité. La raison pour cela est que les démons qui fonctionnent avec une mise à jour peuvent encore utiliser les anciennes bibliothèques après la mise à jour[3]. Pour détecter quels démons peuvent devoir être redémarrés, vous pouvez utiliser le programme **checkrestart** (disponible dans le paquet debian-goodies) ou utiliser cette ligne de commande[4] (en tant que superutilisateur):

À partir de Debian *Jessie*, il est possible d'installer le paquet needrestart qui s'exécutera après chacune mise à niveau par APT et vous demandera de redémarrer les services qui ont été affectés par les mises à jour qui viennent d'être installées. Dans les versions antérieures, vous pouvez exécuter vous-même le programme **checkrestart** (disponible dans le paquet debian-goodies) après la mise à niveau par APT.

[2] Depuis *Etch*.
[3] Bien que les bibliothèques aient été supprimées du système de fichiers, aucun inœud ne sera nettoyé tant qu'un programme a encore un descripteur de fichier pointant dessus.
[4] En fonction de la version de lsof, vous pourriez avoir besoin de remplacer $9 par $8.

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)[5].

Faire passer le système en niveau d'exécution1 (utilisateur seul), puis ensuite au niveau d'exécution3 (multiutilisateur) devrait entraîner le redémarrage de la plupart (si ce n'est tous) des services système. Mais cela n'est pas envisageable si vous exécutez la mise à jour de sécurité depuis une connexion distante (comme SSH) car celle-ci serait alors interrompue.

Apportez le plus grand soin lors des mises à jour de sécurité si vous les réalisez depuis une connexion à distance comme SSH. Une procédure suggérée pour une mise à jour de sécurité qui implique un redémarrage de services est de redémarrer le démon SSH, puis immédiatement de tenter une nouvelle connexion SSH sans interrompre la précédente. Si la connexion échoue, annulez la mise à jour et analysez le problème.

Mise à jour de sécurité du noyau

Assurez-vous tout d'abord que le noyau est géré par le système de gestion des paquets. Si vous l'avez installé en utilisant le système d'installation de Debian3.0 ou de versions précédentes, le noyau *n'est pas* intégré dans le système de gestion des paquets et pourrait être obsolète. Vous pouvez facilement confirmer cela en exécutant:

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

Si le noyau n'est pas géré, vous verrez un message indiquant que le gestionnaire de paquets n'a pas trouvé le fichier associé à un paquet au lieu du message ci-dessus, qui dit que le fichier associé au noyau actuellement en fonctionnement est fourni par le paquet linux-image-2.6.18-4-686. Dans le premier cas, vous devrez installer manuellement un paquet d'image de noyau. L'image exacte du noyau que vous devez installer dépend de l'architecture et de la version de noyau préférée. Une fois fait, vous pourrez gérer les mises à jour de sécurité du noyau comme pour tout autre paquet. Dans tous les cas, notez que les mises à jour du noyau ne seront faites *que* pour la même version du noyau que celui que vous utilisez, c'est-à-dire que **apt** ne va pas mettre à jour automatiquement le noyau de la version2.4 à la version2.6 (ou de la version2.4.26 à la version2.4.27[6]).

Le système d'installation des dernières versions de Debian gérera le noyau sélectionné comme partie du système de gestion des paquets. Vous pouvez vérifier quels noyaux sont installés en exécutant:

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

Pour voir si le noyau doit être mis à jour, exécutez:

```
$ kernfile=`readlink -f /vmlinuz`
$ kernel=`dpkg -S $kernfile | awk -F : '{print $1}'`
$ apt-cache policy $kernel
linux-image-2.6.32-5-686:
  Installé : 2.6.32-35
  Candidat : 2.6.32-35
  Table de version :
 *** 2.6.32-35
```

[5] 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.

[6] Sauf si vous avez installé un métapaquet de noyau comme linux-image-2.6-686 qui va toujours tirer la dernière révision mineure de noyau pour une version de noyau et une architecture donnée.

```
100 /var/lib/dpkg/status
```

Si vous effectuez une mise à jour de sécurité incluant l'image du noyau, vous *devez* redémarrer le système pour que la mise à jour de sécurité soit utile. Sinon, vous utiliserez encore l'ancienne image de noyau (vulnérable).

Si vous devez effectuer un redémarrage du système (à cause d'une mise à jour du noyau), vous devriez vous assurer que le noyau démarrera correctement et que la connectivité réseau sera restaurée, particulièrement si la mise à jour de sécurité est réalisée depuis une connexion à distance comme SSH. Pour le premier point, vous pouvez configurer le chargeur d'amorçage pour redémarrer l'ancien noyau en cas d'échec (pour des informations plus détaillées, veuillez consulter (en anglais) http://www.debian-administration.org/?article=70). Pour le second point, vous devez introduire un script de test de connectivité réseau qui vérifiera si le noyau a lancé le sous-système réseau correctement et qui redémarrera le système si ce n'est pas le cas[7]. Cela devrait éviter des surprises désagréables comme une mise à jour du noyau en réalisant après un redémarrage qu'il n'a pas détecté ou configuré le matériel réseau correctement et que vous devez parcourir une longue distance pour relancer à nouveau le système. Bien sûr, avoir la console série[8] du système connectée à une console ou un serveur de terminal devrait également aider à déboguer à distance les problèmes de redémarrage.

Changer le BIOS (à nouveau)

Vous vous souvenez de la section intitulée « Choisir un mot de passe pour le BIOS »? Et bien, vous devriez maintenant, une fois que vous n'avez plus besoin de démarrer à partir d'un support amovible, changer la configuration par défaut du BIOS pour qu'il ne puisse démarrer *que* depuis le disque dur. Assurez-vous de ne pas perdre le mot de passe BIOS, sinon, en cas de défaillance du disque dur, vous ne pourrez pas retourner dans le BIOS et modifier la configuration pour le récupérer en utilisant, par exemple, un CD.

Un autre moyen moins sécurisé, mais plus pratique est de changer la configuration pour que le système s'amorce depuis le disque dur et, si cela échoue, d'essayer un support amovible. À propos, c'est ainsi fait parce que la plupart des personnes n'utilisent pas le mot de passe BIOS très souvent; il est facilement oublié.

Attribuer un mot de passe à LILO ou GRUB

N'importe qui peut obtenir facilement une invite de commandes superutilisateur et changer les mots de passe en entrant à l'invite d'amorçage

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

Après le changement du mot de passe et le redémarrage du système, la personne a un accès superutilisateur illimité et peut faire tout ce qu'elle veut sur le système. Après cela, vous n'aurez plus d'accès supertilisateur sur la machine, étant donné que vous ne connaîtrez pas le mot de passe.

Pour être sûr que cela ne puisse pas se produire, vous devriez attribuer un mot de passe au démarrage. Vous avez le choix entre un mot de passe global et un mot de passe par image.

Pour LILO, vous avez besoin d'éditer le fichier `/etc/lilo.conf` et ajouter les lignes **`password`** ainsi que **`restricted`** comme dans l'exemple suivant.

[7] Un exemple d'un tel script appelé http://www.debian-administration.org/articles/70/testnet est disponible dans l'article http://www.debian-administration.org/?article=70. Un script de test de connectivité réseau plus élaboré est disponible dans l'article http://www.debian-administration.org/?article=128.

[8] Configurer une console série est en dehors du cadre de ce document, pour plus d'informations, veuillez consulter le http://www.tldp.org/HOWTO/Serial-HOWTO.html et le http://www.tldp.org/HOWTO/Remote-Serial-Console-HOWTO/index.html.

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

Puis, assurez-vous que le fichier de configuration n'est pas lisible par tout le monde pour empêcher des utilisateurs locaux de lire le mot de passe. Une fois terminé, relancez lilo. Omettre la ligne `restricted` entraîne une attente de mot de passe, en dépit des paramètres passés à LILO. Les permissions par défaut pour le fichier `/etc/lilo.conf` accordent au superutilisateur les droits de lecture et d'écriture et permettent un accès en lecture seule pour le groupe de configuration de `lilo.conf`, à savoir root.

Si vous utilisez GRUB plutôt que LILO, éditez `/boot/grub/menu.lst` et ajoutez les deux lignes suivantes en début (en remplaçant, bien sûr, **`hackme`** par le mot de passe désiré). Cela empêche les utilisateurs d'éditer les options de démarrage. **`timeout 3`** indique un délai de 3 secondes avant que **grub** démarre l'option par défaut.

```
timeout 3
  password piratemoi
```

Pour aller plus loin dans le durcissement de l'intégrité du mot de passe, vous pourriez entreposer le mot de passe sous une forme chiffrée. L'utilitaire **grub-md5-crypt** génère un hachage de mot de passe qui est compatible avec l'algorithme du mot de passe GRUB (MD5). Pour indiquer à **grub** qu'un mot de passe au format MD5 va être utilisé, utilisez la directive suivante:

```
timeout 3
  password --md5 $1$T/vfEWUQ$t8xoW.5kp3nbqc1zOwa3W1
```

Le paramètre --md5 a été ajouté pour informer **grub** d'effectuer la procédure d'authentification md5. Le mot de passe fourni est la version MD5 chiffrée de piratemoi. L'utilisation de la méthode MD5 pour le mot de passe est préférable à la méthode précédente dont le mot de passe est en clair. Plus d'informations concernant les mots de passe **grub** sont disponibles dans le paquet grub-doc.

Désactivation de l'invite superutilisateur de l'initramfs

Note: cela s'applique aux noyaux fournis par défaut après Debian 3.1.

Les noyaux Linux 2.6 fournissent un moyen d'accéder à une invite de commande de superutilisateur lors de l'amorçage et qui sera présentée pendant le chargement de l'initramfs en cas d'erreur. C'est pratique pour permettre à l'administrateur d'entrer une invite de commande de secours avec des droits du superutilisateur. Cette invite de commande peut être utilisée pour charger vous-même des modules quand la détection automatique échoue. Ce comportement est celui par défaut pour les initramfs créés par **initramfs-tools**. Le message suivant apparaîtra:

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

Afin de supprimer ce comportement, vous devez configurer l'argument d'amorçage suivant : *panic=0*. Ajoutez-le soit a la variable `GRUB_CMDLINE_LINUX` a `/etc/default/grub` et exécutez**update-grub**, soit à la section append de `/etc/lilo.conf`.

Enlever l'invite superutilisateur du noyau

Note: cela ne s'applique pas aux noyaux fournis par Debian3.1, car le temps d'attente du noyau a été modifié à 0.

Les noyaux Linux 2.4 fournissent un moyen d'accéder à une invite de commandes superutilisateur lors de l'amorçage et qui sera présenté juste après le chargement du système de fichiers cramfs. Un message apparaîtra pour permettre à l'administrateur d'obtenir une invite de commandes interactive avec des droits du superutilisateur, cette invite de commandes peut être utilisée pour charger manuellement des modules quand la détection automatique échoue. Ce comportement est celui par défaut pour `linuxrc` de l'**initrd**. Le message suivant apparaîtra:

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

Pour supprimer ce comportement, vous devez changer `/etc/mkinitrd/mkinitrd.conf` et positionner:

```
# DELAY  Le temps, en seconde que le script linuxrc doit
  # attendre pour permettre à l'utilisateur de l'interrompre
  # avant que le système ne soit lancé
  DELAY=0
```

Puis, régénérez l'image de ramdisk. Vous pouvez faire cela ainsi, par exemple:

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

ou (de préférence):

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

Restreindre les accès aux consoles

Certaines règles de sécurité peuvent forcer les administrateurs à se connecter au système sur une console avec leur identifiant et mot de passe puis devenir superutilisateur (avec **su** ou **sudo**). Cette règle est appliquée sous Debian en éditant les fichiers `/etc/pam.d/login` et `/etc/securetty` lors de l'utilisation de PAM :

`/etc/pam.d/login` [9] active le module pam_securetty.so. Ce module, une fois correctement configuré, interdira la demande de mot de passe quand le superutilisateur essaye de se connecter sur une console non sécurisée, rejetant l'accès à cet utilisateur ;

`securetty`[10] en ajoutant ou supprimant les terminaux depuis lesquels les accès du superutilisateur seront autorisés. Si vous voulez n'autoriser que les accès locaux en console, vous avez alors besoin de *console*, *ttyX*[11] et *vc/X* (si vous utilisez des périphériques *devfs*), vous pouvez vouloir ajouter également *ttySX*[12] si

[9] Dans les versions précédentes de Debian, il fallait modifier `login.defs`, et utiliser la variable CONSOLE qui définit un fichier ou une liste de terminaux sur lesquels la connexion du superutilisateur est autorisée.
[10] Le fichier `/etc/securetty` est un fichier de configuration qui appartient au paquet login.
[11] Ou *ttyvX* pour GNU/FreeBSD et *ttyE0* pour GNU/KNetBSD.
[12] Ou *comX* pour GNU/Hurd, *cuaaX* pour GNU/FreeBSD et *ttyXX* pour GNU/KNetBSD.

vous utilisez une console série pour l'accès local (où X est un nombre entier, vous pouvez vouloir avoir plusieurs instances). La configuration par défaut pour *Wheezy* [13] inclut de nombreux périphériques tty, ports séries, consoles vc ainsi que le serveur X et le périphérique *console*. Vous pouvez ajuster cela en toute sécurité si vous n'utilisez pas tant de consoles. Vous pouvez confirmer les consoles virtuelles et périphériques tty disponibles en vérifiant `/etc/inittab` [14]. Pour plus d'informations sur les périphériques de terminal, veuillez consulter le http://tldp.org/HOWTO/Text-Terminal-HOWTO-6.html (ou la http://www.traduc.org/docs/HOWTO/vf/Text-Terminal-HOWTO.html).

En cas d'utilisation de PAM d'autres changements au processus de login, qui peuvent inclure des restrictions aux utilisateurs et groupes à certains moments, peuvent être configurés dans `/etc/pam.d/login`. Une fonctionnalité intéressante qui peut être désactivée est la possibilité de se connecter avec des mots de passe nuls (vides). Cette fonctionnalité peut être limitée en enlevant *nullok* de la ligne:

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

Restreindre les redémarrages système depuis la console

Si un clavier est attaché au système, tout le monde (oui, *tout le monde*) avec un accès physique au système peut le redémarrer sans se connecter en appuyant simplement sur la combinaison *Ctrl+Alt+Delete* du clavier (le *salut à trois doigts*). Cela peut être en conformité ou non avec vos règles de sécurité.

C'est aggravé dans les environnements où le système d'exploitation est virtualisé. Dans ces environnements, la possibilité s'étend aux utilisateurs ayant accès à la console virtuelle (qui pourrait être accessible par le réseau). Remarquez aussi que, dans ces environnements, cette combinaison est utilisée constamment (pour ouvrir une invite de connexion dans certaines interfaces graphiques de systèmes d'exploitations) et qu'un administrateur pourrait l'envoyer *virtuellement* et forcer un système à redémarrer.

Deux manières permettent de restreindre cela :

- le configurer pour que seuls les utilisateurs *autorisés* puissent redémarrer le système ;
- désactiver complètement cette fonctionnalité.

Si vous désirez restreindre cela, vous devez vérifier le fichier `/etc/inittab` pour que la ligne incluant **`ctrlaltdel`** appelle **shutdown** avec le paramètre **`-a`**.

La valeur par défaut dans Debian inclut ce paramètre :

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

Le paramètre **`-a`**, conformément à la description de la page de manuel shutdown(8), donne la possibilité de permettre à *certains* utilisateurs d'arrêter le système. Pour cela, le fichier `/etc/shutdown.allow` doit être créé et inclure le nom des utilisateurs qui peuvent redémarrer le système. Quand la combinaison du *salut à trois doigts* est exécutée en console, le programme va vérifier si l'un des utilisateurs définis dans ce fichier est connecté. Si aucun d'entre eux ne l'est, **shutdown** ne va *pas* redémarrer le système.

Pour désactiver la combinaison Ctrl+Alt+Del, il suffit de commenter la ligne contenant la définition de *ctrlaltdel* dans `/etc/inittab`.

[13] La configuration par défaut dans *Woody* inclut 12consoles locales tty et vc, ainsi que le périphérique *console*, mais ne permet pas les connexions distantes. Dans *Sarge*, la configuration par défaut fournit 64consoles pour les consoles tty et vc.
[14] Recherchez les appels *getty*.

N'oubliez pas d'exécuter **`init q`** après toute modification du fichier `/etc/inittab` pour qu'elle prenne effet.

Restriction d'utilisation des touches SysRq magiques

Les *touches SysRq magiques* sont des combinaisons de touches qui permettent aux utilisateurs connectés à une console système du noyau Linux de réaliser des commandes de bas niveau. Ces commandes de bas niveau sont envoyées en appuyant simultanément sur *Alt+SysRq* et une touche de commande. La touche SysRq est la même que « Impr écran » sur la plupart des claviers.

Depuis la publication de Etch, les touches SysRq magiques sont activées dans le noyau Linux pour donner certains droits aux utilisateurs en console. Vous pouvez confirmer cela en vérifiant si `/proc/sys/kernel/sysrq` existe et vérifier ses valeurs :

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

La valeur par défaut ci-dessus permet toutes les fonctions SysRq sauf la possibilité d'envoyer des signaux aux processus. Par exemple, elle permet aux utilisateurs connectés en console de remonter tous les systèmes de fichiers en lecture seule, redémarrer le système ou provoquer une panique du noyau. Si toutes les fonctionnalités sont activées, ou sur les anciens noyaux (avant le 2.6.12), la valeur sera simplement 1.

Vous devriez désactiver cette fonctionnalité si l'accès à la console n'est pas restreint aux utilisateurs autorisés : par exemple si la console est connectée à une ligne modem, s'il y a un accès physique facile au système ou s'il est exécuté dans un environnement virtualisé et d'autres utilisateurs ont accès à la console. Pour cela, modifiez `/etc/sysctl.conf` pour ajouter les lignes suivantes :

```
# Désactiver les touches SysRq magiques
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].

Monter correctement les partitions

En montant un système de fichiers `ext` (`ext2`, `ext3` ou `ext4`), différentes options additionnelles sont permises pour l'appel mount ou pour le fichier `/etc/fstab`. Par exemple, ceci est une entrée pour la partition `/tmp`:

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

Vous voyez la différence dans la section des options. L'option `nosuid` ignore complètement les bits setuid et setgid, tandis que `noexec` interdit l'exécution de tout programme sur ce point de montage et `nodev` ignore les fichiers de périphériques. Cela semble bon mais :

- ne s'applique qu'aux systèmes de fichiers `ext2` ou `ext3` ;
- peut être contourné facilement.

L'option `noexec` évite aux binaires d'être exécutés directement mais c'était facilement contournable dans les premières versions du noyau:

```
alex@joker:/tmp# mount | grep tmp
/dev/hda7 on /tmp type ext2 (rw,noexec,nosuid,nodev)
alex@joker:/tmp# ./date
bash: ./date: Permission non accordée
alex@joker:/tmp# /lib/ld-linux.so.2 ./date
dimanche 3 décembre 2000, 17:49:23 (UTC+0100)
```

Les versions plus récentes du noyau gèrent cependant l'option `noexec` correctement:

```
angrist:/tmp# mount | grep /tmp
/dev/hda3 on /tmp type ext3 (rw,noexec,nosuid,nodev)
angrist:/tmp# ./date
bash: ./tmp: Permission non accordée
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
```

Toutefois, de nombreux pirates en herbe utilisent des failles qui essayent de créer et d'exécuter des fichiers dans `/tmp`. S'ils ne sont pas futés, ils tomberont sur un pépin. En d'autres termes, un utilisateur ne peut être abusé en exécutant un binaire compromis (genre cheval de Troie) dans `/tmp` lorsqu'il a accidentellement ajouté `/tmp` dans son PATH.

Soyez aussi vigilant, certains scripts peuvent dépendre du fait que `/tmp` devienne exécutable. Notamment Debconf qui a (avait?) quelques problèmes concernant cela, pour plus d'informations consultez le http://bugs.debian.org/116448.

Ce qui suit est un exemple un plus peu poussé. Prenez note que, bien que `/var` peut être mis à noexec, certains logiciels[15] conservent leurs programmes dans /var. Les mêmes conditions peuvent être appliquées à l'option nosuid.

```
/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
```

Paramétrer /tmp en noexec

Soyez vigilant si vous mettez `/tmp` en noexec et que vous voulez installer de nouveaux logiciels étant donné que certains peuvent l'utiliser pendant l'installation. apt est un programme de ce genre (consultez http://bugs.debian.org/116448) si `APT::ExtractTemplates::TempDir` n'est pas configuré correc-

[15] Cela inclut le gestionnaire de paquet dpkg car les scripts d'installation (pre et post) et de suppression (pre et post) sont en `/var/lib/dpkg/` et aussi Smartlist.

tement (consultez apt-extracttemplates(1)). Vous pouvez paramétrer cette variable dans le fichier `/etc/apt/apt.conf` vers un autre répertoire que `/tmp` et qui aura les droits d'exécution.

Paramétrer /usr en lecture seule

Si vous mettez `/usr` en lecture seule, vous serez dans l'incapacité d'installer de nouveaux paquets sur le système Debian GNU/Linux. Vous devrez, avant tout, la remonter en lecture/écriture, puis installer les nouveaux paquets et enfin la remonter en lecture seule. apt peut être configuré pour lancer des commandes avant et après l'installation de paquets, ainsi vous pouvez avoir envie de le configurer correctement.

Pour réaliser cela, modifiez le fichier `/etc/apt/apt.conf` et ajoutez:

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

Notez que le Post-Invoke peut échouer avec un message d'erreur « /usr busy ». Cela survient principalement lorsque vous utilisez des fichiers lors de la mise à jour et que ces fichiers sont justement mis à jour. Vous pouvez trouver ces programmes en exécutant

```
# lsof +L1
```

Arrêtez ou relancez ces programmes et exécutez la commande de Post-Invoke vous-même. *Attention!* Cela veut dire que vous devrez probablement redémarrer la session X (si vous en faites fonctionner une) à chaque fois que vous faites une mise à jour majeure du système. Vous pourriez aussi vous redemander si paramétrer `/usr` en lecture seule est adapté au système. Consultez également cette http://lists.debian.org/debian-devel/2001/11/threads.html#00212.

Fournir des accès sécurisés aux utilisateurs

Authentification utilisateur: PAM

PAM (Pluggable Authentication Modules) permet aux administrateurs système de choisir comment les applications authentifient les utilisateurs. Remarquez que PAM ne peut rien faire tant qu'une application n'a pas été compilée avec la prise en charge pour PAM. La plupart des applications livrées dans Debian ont cette prise en charge intégrée (Debian n'avait pas de prise en charge pour PAM avant la version 2.2). La configuration actuelle par défaut pour tout service activé avec PAM est d'émuler l'authentification UNIX (consultez `/usr/share/doc/libpam0g/Debian-PAM-MiniPolicy.gz` pour plus d'informations sur la façon dont les services *devraient* fonctionner dans Debian).

Chaque application avec la prise en charge de PAM fournit un fichier de configuration dans `/etc/pam.d` qui peut être utilisé pour modifier son comportement

- quelle fonction de base est utilisée pour l'authentification;
- quelle fonction de base est utilisée pour les sessions;
- comment les vérifications de mots de passe se comportent.

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 vous offre la possibilité de passer en revue plusieurs étapes d'authentification en une seule fois, à l'insu de l'utilisateur. Vous pouvez vous authentifier à une base de données Berkeley et à un fichier `passwd` normal, ainsi l'utilisateur pourra se connecter seulement si l'authentification est correcte des deux côtés. Vous pouvez restreindre beaucoup de choses avec PAM comme vous pouvez laisser libre accès au système. Donc soyez prudent. Une ligne de configuration typique a un champ de contrôle comme deuxième élément. Généralement, il devrait être paramétré sur `requisite` qui retourne un échec de connexion si un module échoue.

Sécurité de mot de passe dans PAM

Vérifiez `/etc/pam.d/common-password`, englobé dans `/etc/pam.d/passwd` [16]. Ce fichier est aussi englobé dans d'autres fichiers de `/etc/pam.d/` pour définir le comportement des mots de passe utilisés dans les sous-systèmes qui donnent accès aux services sur la machine, comme la connexion en console (login), les gestionnaires de connexion graphiques (comme gdm ou lightdm) et la connexion à distance (comme sshd).

Assurez-vous que le module pam_unix.so utilise l'option « sha512 » pour les mots de passe chiffrés. C'est le cas par défaut dans Debian Squeeze.

La ligne avec la définition du module pam_unix devrait ressembler à :

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

Cette définition :

- renforce le chiffrement des mots de passe conservés, en utilisant la fonction de hachage SHA-512 (option *sha512*) ;
- active les vérifications de complexité des mots de passe (option *obscure*) conformément à la définition de la page de manuel pam_unix(8),
- impose une taille minimale de mot de passe (option *min*) à 8 caractères.

Assurez-vous que les mots de passe chiffrés sont utilisés dans les applications PAM, car cela aide à protéger contre les attaques par dictionnaire. L'utilisation du chiffrement permet aussi d'utiliser des mots de passe plus longs que 8 caractères.

Puisque ce module est aussi utilisé pour définir la façon dont les mots de passe sont changés (il est inclus par **chpasswd**), vous pouvez renforcer la sécurité des mots de passe dans le système en installant libpam-cracklib et en introduisant cette définition dans le fichier de configuration `/etc/pam.d/common-password` :

```
# Vérifier que libpam-cracklib soit installé avant sinon vous ne
# pourrez pas vous connecter.
password   required     pam_cracklib.so retry=3 minlen=12 difok=3
password   [success=1 default=ignore]      pam_unix.so obscure minlen=8 sha512 u
```

La première ligne charge le module PAM cracklib, qui fournit la vérification de la sûreté des mots de passe, attend un nouveau mot de passe avec une taille minimale [17] de 12 caractères, une différence d'au

[16] Dans les anciennes versions de Debian, la configuration des modules était définie directement dans `/etc/pam.d/passwd`.

[17] L'option *minlen* n'est pas tout à fait claire, et n'indique pas exactement le nombre de caractères du mot de passe. Un compromis peut être défini entre la complexité et la taille en ajustant les paramètres « credit » de différentes classes de caractères. Pour plus de renseignements, consultez la page de manuel pam_cracklib(8).

moins 3 caractères par rapport à l'ancien et autorise 3 essais. cracklib dépend d'une liste de mots (comme wenglish, wfrench, wbritish, etc.), assurez-vous donc d'en avoir installé une adaptée à votre langue, sinon, cela peut être totalement inutile.

La seconde ligne (utilisant le module pam_unix.so) est la configuration par défaut dans Debian, conformément à la description précédente, hormis pour l'option *use_authok*. L'option `use_authok` est nécessaire si pam_unix.so est empilé après pam_cracklib.so, et est utilisé pour passer le mot de passe du module précédent. Sinon le mot de passe serait demandé deux fois à l'utilisateur.

Pour plus de renseignements sur le réglage de cracklib, consultez la page de manuel pam_cracklib(8) et l'article http://www.deer-run.com/~hal/sysadmin/pam_cracklib.html de Hal Pomeranz.

En activant le module PAM cracklib, vous définissez une règle qui oblige les utilisateurs à utiliser des mots de passe sûrs.

Autrement, les modules PAM peuvent être configurés pour utiliser une authentification à deux facteurs, comme : libpam-barada, libpam-google-authenticator, libpam-oath, libpam-otpw, libpam-poldi, libpam-usb ou libpam-yubico. La configuration de ces modules permettrait d'accéder au système en utilisant des mécanismes d'authentification externes comme des cartes à puce, clefs USB externes ou mots de passe uniques générés par des applications externes exécutées, par exemple, sur le téléphone portable de l'utilisateur.

Veuillez remarquer que ces restrictions s'appliquent à tous les utilisateurs mais *pas* aux changements de mot de passe du superutilisateur. Le superutilisateur pourra définir n'importe quel mot de passe (n'importe quelle taille ou complexité) pour lui-même et les autres, quelque soient les restrictions définies ici.

Contrôle de l'accès utilisateur dans PAM

Afin d'être sûr que le superutilisateur peut se connecter uniquement à partir des terminaux locaux, la ligne suivante doit être activée dans `/etc/pam.d/login`:

```
auth     requisite  pam_securetty.so
```

Puis, vous devez modifier la liste des terminaux sur lesquels la connexion du superutilisateur est autorisée dans le fichier `/etc/securetty` (comme c'est décrit en la section intitulée « Restreindre les accès aux consoles »). Vous pouvez sinon activer le module pam_access et modifier `/etc/security/access.conf` qui permet un contrôle plus général et affiné, mais à qui il manque (malheureusement) des messages de journalisation décents (la journalisation dans PAM n'est pas standard et est un problème particulièrement peu gratifiant à traiter). Nous reviendrons au fichier `access.conf` un peu plus tard.

Limites des utilisateurs dans PAM

La ligne suivante devrait être activée dans `/etc/pam.d/login` pour mettre en place des limites de ressource utilisateur.

```
session  required   pam_limits.so
```

Cela restreint les ressources du système auxquelles les utilisateurs sont autorisées (consultez ci-après la section intitulée « Restreindre l'utilisation des ressources: le fichier limits.conf »). Par exemple, vous pouvez restreindre le nombre de connexions (d'un groupe d'utilisateurs donné ou tout le système), le nombre de processus, la taille de la mémoire, etc.

Contrôle de su dans PAM

Si vous voulez protéger **su**, pour que seules quelques personnes puissent l'utiliser pour devenir superutilisateur sur le système, vous avez besoin de créer un nouveau groupe «wheel» (c'est la meilleure façon, étant donné qu'aucun fichier n'a ces permissions d'attribuées). Ajoutez root et les autres utilisateurs, qui auront la possibilité d'utiliser **su** pour devenir superutilisateur, à ce groupe. Ensuite, ajoutez la ligne suivante dans `/etc/pam.d/su`:

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

Cela permet d'être sûr que seules les personnes du groupe «wheel» pourront utiliser **su** pour devenir superutilisateur. Les autres utilisateurs ne seront pas capables de le devenir. En fait, ils recevront un message de refus s'ils essayent de devenir superutilisateur.

Si vous désirez que seulement certains utilisateurs s'authentifient à un service PAM, il suffit d'utiliser les fichiers où sont stockés les utilisateurs autorisés (ou pas) à se connecter. Imaginons que vous ne vouliez autoriser que l'utilisateur «ref» à se connecter avec **ssh**. Vous le mettez dans `/etc/sshusers-allowed` et écrivez ce qui suit dans `/etc/pam.d/ssh`:

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

Répertoires temporaires dans PAM

Puisqu'il y eu de nombreuses vulnérabilités dites de fichier temporaire non sécurisé, dont thttpd est un exemple (consultez http://www.debian.org/security/2005/dsa-883), libpam-tmpdir est un bon paquet à installer. Tout ce que vous avez à faire est d'ajouter ceci à `/etc/pam.d/common-session` :

```
session    optional     pam_tmpdir.so
```

Une discussion a eu lieu à propos de l'ajout par défaut dans Debian. Consultez http://lists.debian.org/debian-devel/2005/11/msg00297.html pour obtenir plus de renseignements.

Configuration pour les applications PAM non définies

La dernière étape, mais pas la moindre, est de créer le fichier `/etc/pam.d/other` et d'ajouter les lignes suivantes:

```
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
```

Ces lignes vont fournir une bonne configuration par défaut pour toutes les applications qui gèrent PAM (accès refusé par défaut).

Restreindre l'utilisation des ressources: le fichier limits.conf

Vous devriez vraiment jeter un sérieux coup d'œil à ce fichier. Vous pouvez y définir les limites des ressources par utilisateur. Dans d'anciennes versions, ce fichier de configuration était `/etc/limits.conf`, mais dans les nouvelles versions (avec PAM), le fichier de configuration à utiliser devrait être `/etc/security/limits.conf`.

Si vous ne désirez pas restreindre l'utilisation des ressources, *n'importe quel* utilisateur ayant une invite de commandes valable sur le système (ou même un intrus qui aurait compromis le système par un service ou un démon devenu fou) pourra utiliser autant de CPU, de mémoire, de pile, etc. que le système pourra fournir. Ce problème d'*épuisement de ressources* peut être réglé par l'utilisation de PAM.

Il existe un moyen d'ajouter des limites de ressources pour certains interpréteurs de commandes (par exemple, **bash** a **ulimit**, consultez bash(1)), mais comme ils ne fournissent pas tous les mêmes limites et qu'un utilisateur peut changer d'interpréteur (consultez chsh(1)), il est préférable de placer ces limites dans les modules PAM ainsi elles s'appliqueront quel que soit l'interpréteur de commandes utilisé et également aux modules PAM qui ne sont pas orientés interpréteur.

Les limites de ressources sont imposées par le noyau, mais elles doivent être configurées par le fichier `limits.conf` et la configuration PAM des différents services doit charger le module PAM approprié. Vous pouvez vérifier quels services imposent des limites en exécutant:

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

Habituellement, `login`, `ssh` et les gestionnaires de session graphique (`gdm`, `kdm` ou `xdm`) devraient imposer des limites aux utilisateurs, mais vous pouvez vouloir faire cela dans d'autres fichiers de configuration de PAM, comme `cron`, pour empêcher les démons système d'accaparer toutes les ressources système..

Les paramètres de limites spécifiques que vous pouvez vouloir imposer dépendent des ressources du système, c'est l'une des principales raisons pour lesquelles aucune limite n'est imposée dans l'installation par défaut.

Par exemple, l'exemple de configuration ci-dessous impose une limite de 100 processus par utilisateur (pour empêcher les *bombes de fork*) ainsi qu'une limite de 10 Mo de mémoire par processus et une limite de 10 connexions simultanées. Les utilisateurs du groupe `adm` ont des limites supérieures et peuvent créer des fichiers core s'ils le désirent (c'est simplement une limite *soft* (soft)).

```
*               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
```

Voici les limites qu'un utilisateur standard (y compris les démons système) aurait:

```
$ 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
```

Et voici les limites d'un utilisateur administratif:

```
$ 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
```

Pour plus d'informations, consultez:

- 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].
- l'article http://seifried.org/security/os/linux/20020324-securing-linux-step-by-step.html pour la section *Limiting users overview* ;
- le http://seifried.org/lasg/users/ pour la section *Limiting and monitoring users*.

Actions de connexion de l'utilisateur: modification de /etc/login.defs

La prochaine étape est d'éditer les configuration et action de base lors de la connexion de l'utilisateur. Notez que ce fichier ne fait pas partie de la configuration PAM, c'est un fichier de configuration qui est pris en compte par les programmes login et **su**, il n'est pas logique de l'adapter aux cas pour lesquels ni l'un

ni l'autre des programmes n'est appelé au moins indirectement (le programme **getty** qui gère les consoles et offre l'invite de connexion initiale appelle *bien* **login**).

```
FAILLOG_ENAB        yes
```

Si vous activez cette variable, les connexions échouées seront enregistrées dans un journal. Il est important d'en garder une trace pour repérer si quelqu'un tente une attaque par force brute.

```
LOG_UNKFAIL_ENAB    no
```

En configurant cette variable à « yes », les noms d'utilisateur seront enregistrés en cas d'échec de connexion. Laisser la configuration à « no » (par défaut) est plus prudent, puisque sinon, les mots de passe d'utilisateurs pourraient être enregistrés par erreur (si un utilisateur fait une faute de frappe et entre le mot de passe à la place de l'identifiant). Si vous configurez à « yes », assurez-vous que les journaux ont les droits adéquats (640 par exemple, avec une configuration de groupe adéquate comme adm).

```
SYSLOG_SU_ENAB      yes
```

Cela va activer l'écriture dans les journaux de `syslog` des tentatives de **su**. Plutôt important sur des machines sérieuses, mais notez que cela peut aussi bien être à la base de problèmes de respect de la vie privée.

```
SYSLOG_SG_ENAB      yes
```

La même chose que SYSLOG_SU_ENAB, mais s'applique au programme **sg**.

```
ENCRYPT_METHOD  SHA512
```

Comme mentionné ci-dessus, les mots de passe chiffrés réduisent considérablement le problème des attaques par dictionnaire étant donné que vous pouvez utiliser des mots de passe plus longs. Cette définition doit être cohérente avec la valeur définie dans `/etc/pam.d/common-password`.

Actions de connexion de l'utilisateur: modification de `/etc/pam.d/login`

Le fichier de configuration de connexion peut être ajusté pour implémenter une politique plus stricte. Par exemple, la configuration par défaut peut être modifiée pour augmenter le délai entre les invites de connexion. La configuration par défaut définit à 3 secondes le délai :

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

Augmenter la valeur de *delay* à une valeur suffisamment grande permet de rendre plus difficiles les tentatives de connexion en utilisant la force brute. Si un mauvais mot de passe est fourni, le pirate potentiel (ou le simple utilisateur!) doit attendre plus longtemps avant d'obtenir une nouvelle invite de connexion, ce qui prend pas mal de temps quand vous testez des mots de passe. Par exemple, avec *delay=10000000*, les utilisateurs devront attendre 10 secondes s'ils ont tapé un mauvais mot de passe.

Ce fichier permet aussi de définir le message présenté aux utilisateurs avant de se connecter. C'est désactivé par défaut, comme ci-dessous :

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

Si la politique de sécurité l'exige, ce fichier peut être utilisé pour montrer un message standard indiquant que l'accès au système est restreint et que l'accès des utilisateurs est journalisé. Ce type de déclaration peut être nécessaire dans certains environnements et juridictions. Pour l'activer, ajoutez simplement les renseignements nécessaires dans le fichier `/etc/issue` [18] et décommentez la ligne activant le module pam_issue.so dans `/etc/pam.d/login`. Dans ce fichier, vous pouvez aussi activer des fonctionnalités supplémentaires qui pourraient être pertinentes à appliquer aux politiques de sécurité locales comme :

- la définition de règles accordant l'accès à certains utilisateurs suivant l'heure, en activant le module *pam_time.so* et en configurant `/etc/security/time.conf` en conséquence (désactivée par défaut) ;
- la définition de sessions de connexion pour utiliser les limitations aux utilisateurs conformément à la définition de `/etc/security/limits.conf` (activée par défaut) ;
- la présentation à l'utilisateur des renseignements sur la précédente connexion (activée par défaut) ;
- l'affichage d'un message (`/etc/motd` et `/run/motd.dynamic`) aux utilisateurs après la connexion (activée par défaut) ;

Restreindre le FTP: éditer `/etc/ftpusers`

Ce fichier contient une liste d'utilisateurs qui ne sont pas autorisés à se connecter à l'hôte en utilisant FTP. Utilisez uniquement ce fichier si vous voulez réellement autoriser le FTP (qui n'est, en général, pas recommandé car il utilise des mots de passe en clair). Si le démon gère PAM, cela peut être utilisé pour permettre ou refuser certains services aux utilisateurs.

FIXME (bogue) : Est-ce un bogue que le fichier par défaut `ftpusers` dans Debian ne contienne *pas* tous les utilisateurs d'administration (dans base-passwd) ?

Un moyen pratique d'ajouter tous les comptes système à `/etc/ftpusers` est d'exécuter

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

Utilisation de su

Si vous avez réellement besoin que des utilisateurs deviennent superutilisateur sur le système, par exemple pour installer des paquets ou ajouter des utilisateurs, vous pouvez utiliser la commande **su** pour changer d'identité. Vous devriez essayer d'éviter toute connexion en tant que superutilisateur et d'utiliser à la place **su**. En réalité, la meilleure solution est de supprimer **su** et de changer pour le mécanisme **sudo** qui a une logique plus large et plus de fonctionnalités que **su**. Cependant, **su** est plus commun étant donné qu'il est utilisé sur beaucoup d'autres UNIX.

Utilisation de sudo

sudo autorise l'utilisateur à exécuter des commandes définies sous l'identité d'un autre utilisateur, même en tant que superutilisateur. Si l'utilisateur est ajouté à `/etc/sudoers` et est authentifié correctement, il est capable de lancer des commandes qui ont été définies dans `/etc/sudoers`. Les infractions, telles

[18] Le contenu par défaut de ce fichier fournit des renseignements sur le système d'exploitation et la version utilisée par le système, que vous pourriez ne pas vouloir dévoiler aux utilisateurs anonymes.

que les mots de passe incorrects ou les tentatives de lancement d'un programme pour lequel vous n'avez pas les permissions, sont logguées et envoyées au superutilisateur.

Désactiver des accès d'administration à distance

Vous devriez également modifier `/etc/security/access.conf` pour désactiver la connexion d'administration à distance. Ainsi, les utilisateurs doivent exécuter **su** (ou **sudo**) pour utiliser des pouvoirs administratifs et ainsi la trace d'audit appropriée sera toujours générée.

Vous devez ajouter la ligne suivante à `/etc/security/access.conf`, le fichier de configuration par défaut Debian contient une ligne d'exemple commentée:

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

Rappelez-vous d'activer le module pam_access pour chaque service (ou configuration par défaut) dans `/etc/pam.d/` si vous voulez que vos modifications dans `/etc/security/access.conf` soient prises en compte.

Restriction des utilisateurs

Parfois, vous pensez avoir besoin d'utilisateurs créés dans le système local de façon à fournir un service donné (service courrier POP3 ou FTP). Avant tout, rappelez-vous que l'implémentation PAM dans Debian GNU/Linux vous autorise à valider les utilisateurs avec une grande variété de répertoires de services externes (radius, LDAP, etc.) fournis par les paquets libpam.

Si des utilisateurs doivent être créés et que le système est accessible à distance, prenez en compte que des utilisateurs pourront se connecter au système. Cela peut être corrigé en donnant aux utilisateurs un interpréteur de commandes vide (`/dev/null`) (qui doit être dans `/etc/shells`). Si vous voulez autoriser les utilisateurs à accéder au système mais limiter leurs mouvements, vous pouvez utiliser le fichier `/bin/rbash`, ce qui est équivalent à l'ajout de l'option `-r` dans bash (consultez *INTERPRÉTEUR RESTREINT* dans bash(1)). Veuillez noter que même avec un interpréteur de commandes restreint, un utilisateur ayant accès à un programme interactif (qui peut permettre l'exécution d'un sous-interpréteur) peut être capable de passer outre les limites de l'interpréteur de commandes.

Debian fournit actuellement dans la version unstable le module `pam_chroot` (dans le paquet libpam-chroot) (et il pourrait être inclus dans les prochaines versions stables). Une alternative à celui-ci est de **chroot**er le service qui fournit la connexion à distance (**ssh**, **telnet**). [19]

Si vous voulez restreindre *quand* les utilisateurs peuvent accéder au système, vous devrez personnaliser `/etc/security/access.conf` en fonction de vos besoins.

Des informations sur la façon de **chroot**er des utilisateurs accédant au système par le service **ssh** sont décrites dans la section intitulée « Environnement de chroot pour SSH »

Audit d'utilisateur

Si vous êtes vraiment paranoïaque, vous pourriez configurer l'environnement pour superviser ce que les utilisateurs font sur le système. Cette section présente quelques conseils avec différents utilitaires que vous pouvez utiliser.

[19] libpam-chroot n'a pas encore été testé en profondeur, il fonctionne pour **login**, mais il est possible qu'il ne soit pas facile de mettre en place l'environnement pour d'autres programmes.

Audit d'entrée et sortie avec script

Vous pouvez utiliser la commande **script** pour surveiller à la fois ce que les utilisateurs exécutent et les résultats de leurs commandes. Vous ne pouvez pas configurer **script** comme un interpréteur de commandes (même si vous l'ajoutez à `/etc/shells`). Mais vous pouvez faire en sorte que le fichier d'initialisation de l'interpréteur de commandes exécute les commandes suivantes:

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

Bien sûr, si vous faites cela pour tout le système, cela veut dire que l'interpréteur ne continuerait pas à lire les fichiers d'initialisation personnels (car l'interpréteur sera écrasé par **script**). Une solution est de le faire dans les fichiers d'initialisation de l'utilisateur (mais l'utilisateur pourrait alors l'enlever, consultez les commentaires sur cela ci-dessous).

Vous devez également configurer les fichiers dans le répertoire d'audit (dans l'exemple `/var/log/sessions/`) pour que les utilisateurs puissent y écrire, mais pas supprimer le fichier. Cela pourrait être fait, par exemple, en créant les fichiers de session d'utilisateur en avance et en positionnant l'option *append-only* («append-only») en utilisant **chattr**.

Une alternative utile pour les administrateurs système, qui inclut des informations de date, serait:

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

Utiliser le fichier d'historique de l'interpréteur de commandes

Si vous voulez passer en revue ce que les utilisateurs entrent dans l'interpréteur de commandes (mais sans voir le résultat), vous pouvez configurer un `/etc/profile` pour tout le système qui configure l'environnement pour que toutes les commandes soient enregistrées dans le fichier d'historique. La configuration pour tout le système doit être réalisée de telle façon que les utilisateurs ne puissent pas enlever les capacités d'audit de leur interpréteur de commandes. C'est plutôt spécifique à l'interpréteur de commandes, donc assurez-vous que tous les utilisateurs utilisent un interpréteur de commandes qui le permet.

Par exemple, pour bash, le fichier `/etc/profile` pourrait être paramétré ainsi [20]:

```
HISTFILE=~/.bash_history
HISTSIZE=10000
HISTFILESIZE=999999
# Empêcher les utilisateurs d'entrer des commandes qui seraient
# ignorées dans le fichier d'historique
HISTIGNORE=""
HISTCONTROL=""
readonly HISTFILE
readonly HISTSIZE
readonly HISTFILESIZE
readonly HISTIGNORE
readonly HISTCONTROL
export HISTFILE HISTSIZE HISTFILESIZE HISTIGNORE HISTCONTROL
```

[20] Configurer HISTSIZE à une très grande valeur peut poser des problèmes avec certains interpréteur de commandes car l'historique est gardé en mémoire pour la session de chaque utilisateur. Il peut être plus prudent de positionner cela à une valeur assez élevée et de sauvegarder les fichiers d'historique des utilisateurs (si vous avez besoin de tout l'historique de l'utilisateur pour une raison ou une autre).

Afin que cela fonctionne, l'utilisateur doit être seulement capable d'ajouter des informations au fichier `.bash_history`. Vous devez *aussi* positionner l'attribut *append-only* en utilisant le programme **chattr** sur `.bash_history` pour tous les utilisateurs. [21]

Notez que vous pouvez introduire la configuration ci-dessus dans le fichier utilisateur `.profile`. Mais alors vous devriez configurer les permissions correctement de façon à empêcher à l'utilisateur de modifier ce fichier. Cela inclut: les répertoires personnels de l'utilisateur ne doivent *pas* appartenir à l'utilisateur (sinon, il pourrait supprimer le fichier), mais en même temps lui permettre de lire le fichier de configuration `.profile` et d'écrire dans `.bash_history`. Il serait bien de configurer l'attribut *immuable* (également en utilisant **chattr**) pour le `.profile` aussi si vous procédez ainsi.

Audit utilisateur complet avec utilitaires de comptabilité

L'exemple précédent est une manière simple de configurer l'audit utilisateur, mais qui peut ne pas être utile pour des systèmes complexes ou pour ceux dans lesquels les utilisateurs ne peuvent pas exécuter d'interpréteur de commande du tout (ou exclusivement). Si c'est le cas, vous devrez examiner acct, les utilitaires de comptabilité. Ces utilitaires archiveront toutes les commandes exécutées par les utilisateurs ou processus du système au détriment de l'espace disque.

Lors de l'activation de la comptabilité, toutes les informations sur les processus et utilisateurs sont conservées dans `/var/account/`, plus spécifiquement dans le fichier `pacct`. Le paquet de comptabilité inclut certains outils (**sa** et **ac**) afin d'analyser ces données.

Autres méthodes d'audit utilisateur

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 [22] and dump the output into a file. Other useful program is snoopy (see also github: https://github.com/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`).

Inspection des profils utilisateurs

Si vous désirez *voir* ce que font vraiment les utilisateurs, comme l'heure à laquelle ils se connectent, vous pouvez utiliser la base de données `wtmp` qui contient toutes les informations concernant les connexions. Ce fichier peut être employé avec plusieurs utilitaires, parmi eux **sac** peut sortir un profil de chaque utilisateur montrant dans quel créneau horaire il se connecte habituellement au système.

Dans le cas où vous avez la comptabilité activée, vous pouvez également utiliser les outils qu'elle fournit pour déterminer quand les utilisateurs accèdent au système et ce qu'ils exécutent.

Positionner des umasks aux utilisateurs

En fonction de la politique d'utilisateur, vous pourriez modifier la façon dont les renseignements sont partagés entre utilisateurs, c'est-à-dire quels sont les droits de nouveaux fichiers par défaut créés par les utilisateurs.

Le paramètre `umask` par défaut de Debian est *022*, cela signifie que les fichiers (et les répertoires) peuvent être lus et accédés par le groupe de l'utilisateur et par tout autre utilisateur du système. Cette définition est configurée dans le fichier de configuration normalisé `/etc/profile` utilisé par tous les interpréteurs de commandes.

[21] Sans l'attribut append-only les utilisateurs seraient capables de vider le contenu du fichier des historiques avec `.bash_history`.
[22] Ttys are spawned for local logins and remote logins through ssh and telnet

Si la valeur par défaut de Debian est trop permissive pour le système, vous devrez changer ce paramètre umask pour tous les interpréteurs de commandes. Parmi les configurations plus restrictives d'umask, 027 (pas d'accès permis aux nouveaux fichiers pour le groupe *other*, c'est-à-dire aux autres utilisateur du système) ou 077 (pas d'accès permis aux nouveaux fichiers pour les membres du groupe de l'utilisateur) peuvent être utilisés. Debian (par défaut[23]) crée un groupe par utilisateur de telle sorte que seul l'utilisateur soit inclus dans son groupe. Par conséquent, 027 et 077 sont équivalents car le groupe de l'utilisateur ne contient que l'utilisateur lui-même.

Cette modification est configurée en définissant un réglage correct de `umask` pour tous les utilisateurs. Vous pouvez modifier cela en introduisant un appel **umask** dans les fichiers de configuration de l'interpréteur de commandes : `/etc/profile` (source par tous les interpréteurs de commandes compatibles Bourne), `/etc/csh.cshrc`, `/etc/csh.login`, `/etc/zshrc` et probablement d'autres (en fonction des interpréteurs de commandes installés sur le système). Vous pouvez aussi modifier le réglage de `UMASK` dans `/etc/login.defs`. De toutes celles-là, la dernière chargée par l'interpréteur de commandes est prioritaire. L'ordre est : la configuration système par défaut pour l'interpréteur de l'utilisateur (c'est-à-dire `/etc/profile` et les autres fichiers de configuration globaux du système) et ensuite ceux de l'utilisateur (ses `~/.profile`, `~/.bash_profile`, etc.). Certains interpréteurs, cependant, peuvent être exécutés avec une valeur *nologin* avec laquelle certains de ces fichiers pourraient être sautés. Consultez la page de manuel de l'interpréteur pour obtenir de plus amples renseignements.

Pour les connexions qui utilisent **login**, la définition de `UMASK` de `/etc/login.defs` est utilisée avant toutes les autres. Cependant, cette valeur ne s'applique pas aux programmes exécutés par l'utilisateur qui n'utilisent pas **login** comme ceux exécutés à travers **su**, **cron** ou **ssh**.

N'oubliez pas de vérifier et éventuellement modifier les fichiers de configuration utilisateur sous `/etc/skel/` car ce sont ceux qui seront utilisés par défaut quand ils sont créés avec la commande **adduser**. Les fichiers de configuration utilisateur Debian par défaut ne contiennent pas d'appel **umask** mais s'il y en a dans n'importe quel fichier de configuration utilisateur, les utilisateurs nouvellement créés pourraient avoir une valeur différente.

Notez, cependant, que les utilisateurs peuvent modifier leur propre paramètre `umask` s'ils le désirent, le rendant plus permissif ou plus restrictif, en modifiant leurs propres fichiers de configuration utilisateur.

Le paquet libpam-umask règle l'`umask` par défaut utilisant PAM. Après l'installation du paquet, ajoutez ceci à `/etc/pam.d/common-session`:

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

Enfin, vous pourriez envisager de modifier l'umask par défaut du superutilisateur à 022 (tel que défini dans `/root/.bashrc`) à une valeur plus restrictive. Cela évitera à l'administrateur système de laisser fuir par inadvertance des fichiers sensibles lorsqu'il travaille en tant que superutilisateur dans des répertoires lisibles par tous (comme `/tmp`) et en les rendant lisibles aux autres utilisateurs.

Limiter ce que les utilisateurs peuvent voir et accéder

FIXME: Besoin de contenu. Indiquer les conséquences de changement des permissions des paquets lors d'une mise à jour (un administrateur aussi paranoïaque que cela devrait **chroot**er ses utilisateurs au passage) s'il n'utilise pas **dpkg-statoverride**.

Si vous avez besoin d'accorder aux utilisateurs un accès au système avec un interpréteur de commandes, réfléchissez-y très soigneusement. Un utilisateur peut, par défaut à moins d'être dans un environnement

[23] Tel que défini dans `/etc/adduser.conf` (USERGROUPS=yes). Vous pouvez modifier ce comportement en configurant cette valeur à « no », bien que ce ne soit pas recommandé.

extrêmement restreint (comme une prison `chroot`), récupérer un assez grand nombre d'informations concernant le système, y compris :

- certains fichiers de configuration dans `/etc`. Cependant, les permissions par défaut de Debian pour certains fichiers sensibles (qui peuvent, par exemple, contenir des mots de passe) empêcheront l'accès à des informations critiques. Pour voir quels fichiers ne sont accessibles que par le superutilisateur, exécutez par exemple `find /etc -type f -a -perm 600 -a -uid 0` en tant que superutilisateur ;

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

  en tant que superutilisateur

- vos paquets installés, soit en consultant la base de données des paquets, soit dans le répertoire `/usr/share/doc`, soit en devinant en regardant les binaires et bibliothèques installés sur le système ;

- certains fichiers journaux dans `/var/log`. Notez également que quelques fichiers journaux ne sont accessibles qu'au superutilisateur et au groupe `adm` (essayez

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

) et certains ne sont même disponibles que pour le superutilisateur (essayez

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

).

Que peut voir un utilisateur dans le système? Probablement un assez grand nombre de choses, essayez ceci (prenez une profonde respiration):

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

La liste des fichiers qu'un utilisateur peut *voir* et des répertoires auxquels il a accès est affichée.

Limiter l'accès aux informations d'autres utilisateurs

Si vous accordez toujours un accès d'interpréteur de commandes aux utilisateurs, vous pouvez vouloir limiter les informations qu'ils peuvent voir des autres utilisateurs. Les utilisateurs ayant un accès d'interpréteur de commandes ont tendance à créer un grand nombre de fichiers dans leur répertoire $HOME: boîtes aux lettres, documents personnels, configuration des applications X/GNOME/KDE, etc.

Sous Debian, chaque utilisateur est créé avec un groupe associé et aucun utilisateur n'appartient au groupe d'un autre utilisateur. Il s'agit du comportement par défaut: quand un compte d'utilisateur est créé, un groupe du même nom est créé et l'utilisateur lui est attribué. Cela évite le concept d'un groupe *users* qui peut rendre plus difficile pour les utilisateurs de cacher des informations aux autres utilisateurs.

Cependant, les répertoires `$HOME` des utilisateurs sont créés avec les permissions 0755 (lisible par le groupe et par tout le monde). Les permissions de groupe ne sont pas un problème car seul l'utilisateur appartient au groupe, cependant les permissions pour les autres peuvent être (ou non) un problème selon vos règles locales.

Vous pouvez changer ce comportement pour que la création d'utilisateur fournisse des permissions sur `$HOME` différentes. Pour changer ce comportement pour les *nouveaux* utilisateurs quand ils seront créés,

changez *DIR_MODE* dans le fichier de configuration `/etc/adduser.conf` à 0750 (pas d'accès en lecture pour tout le monde).

Les utilisateurs peuvent toujours partager des informations, mais pas directement dans leur répertoire `$HOME` à moins qu'ils ne changent les permissions de celui-ci.

Notez que désactiver les répertoires utilisateur lisibles par tout le monde empêchera les utilisateurs de créer leurs pages personnelles dans le répertoire `~/public_html` car le serveur web ne pourra pas lire un composant du chemin — leur répertoire `$HOME`. Si vous voulez permettre aux utilisateurs de publier des pages HTML dans leur `~/public_html`, changez *DIR_MODE* en 0751. Cela permettra au serveur web d'accéder à ce répertoire (qui devrait lui-même avoir le mode 0755) et de fournir le contenu publié par les utilisateurs. Bien sûr, nous ne parlons ici que d'une configuration par défaut; les utilisateurs peuvent généralement ajuster les permissions de leurs fichiers comme ils le désirent, ou vous pouvez conserver le contenu destiné au web dans un emplacement séparé qui n'est pas un sous-répertoire du répertoire `$HOME` de chaque utilisateur.

Générer des mots de passe utilisateur

Il y a plusieurs cas dans lesquels un utilisateur a besoin de créer un grand nombre de comptes utilisateur et de fournir des mots de passe pour tous ceux-ci. Bien sûr, l'administrateur peut facilement positionner le mot de passe pour être le même que le nom du compte utilisateur, mais cela n'est pas très conseillé sur le plan de la sécurité. Une meilleure approche est d'utiliser un programme de génération de mots de passe. Debian fournit les paquets makepasswd, apg et pwgen qui contiennent des programmes (dont le nom est le même que celui du paquet) qui peuvent être utilisés dans ce but. **makepasswd** génère des mots de passe vraiment aléatoires avec un accent sur la sécurité plus que la prononçabilité tandis que **pwgen** essaie de créer des mots de passe sans signification, mais prononçables (bien sûr, cela dépend de votre langue maternelle). **apg** dispose d'algorithmes pour les deux (il y a une version client/serveur pour ce programme, mais elle n'est pas incluse dans le paquet Debian).

Passwd ne permet pas une attribution non interactive des mots de passe (car il utilise un accès direct au terminal tty). Si vous désirez changer des mots de passe lors de la création d'un grand nombre d'utilisateurs, vous pouvez les créer en utilisant **adduser** avec l'option `--disabled-login`, puis utiliser **usermod** ou **chpasswd** [24] (tous les deux dans le paquet **passwd**, ils sont donc déjà installés). Si vous voulez utiliser un fichier avec toutes les informations pour créer les utilisateurs comme un processus batch, il sera probablement préférable d'utiliser **newusers**.

Vérifier les mots de passe utilisateur

Les mots de passe des utilisateurs peuvent parfois devenir le *maillon faible* de la sécurité d'un système donné. Cela provient du fait que quelques utilisateurs choisissent des mots de passe faibles pour leur compte (et plus il y a d'utilisateurs, plus grandes sont les chances que cela se produise). Même si vous mettez en place des vérifications avec le module PAM cracklib et les limitations sur les mots de passe comme décrites dans la section intitulée « Authentification utilisateur: PAM », les utilisateurs pourront toujours utiliser des mots de passe faibles. Comme l'accès utilisateur peut inclure un accès à une invite de commandes à distance (en espérant que ce soit avec **ssh**), il est important de rendre les mots de passe aussi difficile à deviner que possible pour les attaquants à distance, particulièrement s'ils ont pu récupérer des informations importantes comme les noms d'utilisateur ou même les fichiers `passwd` et `shadow` eux-mêmes.

[24] **chpasswd** ne sait pas gérer la génération de mots de passe MD5, il faut donc lui donner le mot de passe sous sa forme chiffrée avant de l'utiliser avec l'option

`-e`

.

Un administrateur système doit, suivant le nombre d'utilisateurs, vérifier si les mots de passe sont cohérents avec la règle locale de sécurité. Comment vérifier? Essayez de les casser comme le ferait un attaquant s'il avait accès aux mots de passe hachés (le fichier `/etc/shadow`).

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.

Déconnecter les utilisateurs inactifs (idle)

L'inactivité des utilisateurs pose habituellement un problème de sécurité, un utilisateur peut être inactif parce qu'il est parti déjeuner ou parce qu'une connexion à distance s'est bloquée et n'a pas été rétablie. Quelqu'en soit la raison, les utilisateurs inactifs peuvent amener à une compromission:

- car la console de l'utilisateur peut être débloquée et peut être accédée par un intrus ;
- car un attaquant peut être capable de se rattacher lui-même à une connexion réseau fermée et envoyer des commandes à l'invite de commandes distante (c'est assez facile si l'invite de commandes distante n'est pas chiffrée comme avec **telnet**).

Certains systèmes à distance ont même été compromis à travers un **screen** inactif (et détaché).

La déconnexion automatique des utilisateurs inactifs est habituellement une partie qui doit être imposée par les règles de sécurité locales. Plusieurs moyens existent pour cela:

- si bash est l'interpréteur de commandes de l'utilisateur, un administrateur système peut positionner une valeur `TMOUT` par défaut (consultez bash(1)) qui entraînera la déconnexion automatique des utilisateurs distants inactifs. Notez que cela doit être configuré avec l'option `-o` ou les utilisateurs pourront la changer (ou la désactiver) ;
- installez timeoutd et configurez `/etc/timeouts` selon vos règles de sécurité locales. Le démon regardera les utilisateurs inactifs et mettra un terme à leur invite de commandes en fonction ;
- installez autolog et configurez-le pour enlever les utilisateurs inactifs.

Les démons **timeoutd** et **autolog** sont les méthodes préférées car, après tout, les utilisateurs peuvent changer d'interpréteur de commandes par défaut ou peuvent, après avoir exécuté leur interpréteur de commandes par défaut, basculer sur un autre interpréteur de commandes (non contrôlé).

Utilisation de tcpwrappers

L'encapsulation TCP a été développée quand il n'y avait pas de réels filtres de paquets disponibles et que les contrôles d'accès étaient nécessaires. Toutefois, ils sont toujours très intéressants et utiles. L'encapsulation TCP vous permet d'autoriser ou de refuser un service à un hôte ou à un domaine et de définir une règle par défaut pour les autorisations et les refus (toutes réalisées au niveau applicatif). Pour plus de détails, jetez un œil à hosts_access(5).

De nombreux services installés dans Debian sont soit:

- lancés par le service tcpwrapper (`tcpd`) ;
- compilés avec la prise en charge de libwrapper.

D'un côté, pour des services configurés dans `/etc/inetd.conf`, cela comprend **telnet**, **ftp**, **netbios**, **swat** et **finger**), vous observerez que le fichier de configuration exécute avant tout **/usr/sbin/tcpd**. D'un

autre côté, même si un service n'est pas lancé par le super démon **inetd**, il peut être compilé avec la prise en charge pour les règles d'encapsulation TCP. Les services suivant sont compilés avec prise en charge d'encapsulation TCP dans Debian : **ssh**, **portmap**, **in.talk**, **rpc.statd**, **rpc.mountd**, **gdm**, **oaf** (le démon d'activation GNOME), **nessus** et beaucoup d'autres.

Pour voir quels paquets utilisent tcpwrappers [25], essayez :

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

Tenez compte de cela quand vous utilisez **tcpdchk** (un vérificateur très utile de règles et syntaxe de fichier de configuration d'encapsulation TCP). Quand vous pouvez ajouter des services indépendants (qui sont liés à la bibliothèque d'encapsulation) dans les fichiers `host.deny` et `hosts.allow`, **tcpdchk** vous informera qu'il ne peut pas trouver les services mentionnés étant donné qu'il les cherche dans `/etc/inetd.conf` (la page de manuel n'est pas totalement précise ici).

À présent, voici une petite astuce et probablement le plus petit système de détection d'intrusions disponible. Généralement, vous devriez disposer d'une politique correcte concernant le pare-feu en première ligne, puis disposer de l'encapsulation TCP en seconde ligne de défense. Un petit truc est de mettre en place une commande SPAWN [26] dans /etc/hosts.deny qui enverra un courrier au superutilisateur quand un service refusé déclenche l'encapsulation :

```
ALL: ALL: SPAWN ( \
  echo -e "\n\
  Encapsulation TCP \: Connexion refusée\n\
  Par \: $(uname -n)\n\
  Processus \: %d (pid %p)\n\
  Utilisateur \: %u\n\
  Hôte \: %c\n\
  Date \: $(date)\n\
" | /usr/bin/mail -s "Connexion à %d bloquée" root) &
```

Attention: L'exemple ci-dessus peut-être facilement sujet à une attaque par déni de service en soumettant énormément de connexions dans une période très courte. De nombreux courriers signifient de nombreuses E/S en envoyant uniquement quelques paquets.

L'importance des journaux et des alertes

Il est facile de voir que le traitement de journaux et alertes est un problème sérieux sur un système sécurisé. Supposons qu'un système est parfaitement configuré et sécurisé à 99 %. Si l'attaque représentant le 1 % vient à arriver et qu'il n'y a pas de mesures de sécurité mises en place pour, dans un premier temps, détecter cela et, dans un deuxième temps, lancer l'alerte, le système n'est pas sécurisé du tout.

Debian GNU/Linux fournit quelques outils pour effectuer des analyses de journaux, notamment swatch[27], logcheck ou log-analysis (tous ont besoin d'être personnalisés pour enlever les choses non nécessaires des comptes-rendus). Il peut être également utile, si le système est proche, d'avoir les journaux du système affichés sur une console virtuelle. C'est utile car vous pouvez (de loin) voir si le système se comporte correctement. Le fichier `/etc/syslog.conf` de Debian est fourni avec une configuration commentée

[25] Pour les anciennes versions de Debian, vous pourriez devoir utiliser :

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

[26] Assurez-vous d'utiliser des majuscules sinon *spawn* ne fonctionnera pas.

[27] Il y a un très bon article sur celui-ci écrit par http://www.spitzner.net/swatch.html.

par défaut; pour l'activer, décommenter les lignes et redémarrez **syslogd** (**`/etc/init.d/syslogd restart`**:

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

Pour colorer les journaux, vous pouvez jeter un œil à colorize, ccze ou glark. Une grande partie de l'analyse des journaux ne peut pas être couverte ici, une bonne ressource d'informations est disponible dans les livres comme http://books.google.com/books?id=UyktqN6GnWEC. Dans tous les cas, même des outils automatiques ne peuvent rivaliser avec le meilleur outil d'analyse: votre cerveau.

Utiliser et personnaliser logcheck

Le paquet **logcheck** dans Debian est divisé en trois paquets logcheck (le programme principal), logcheck-database (une base de données d'expressions rationnelles pour le programme) et logtail (affiche les lignes du journal qui n'ont pas encore été lues). Le comportement par défaut sous Debian (dans `/etc/cron.d/logcheck`) est que **logcheck** est exécuté toutes les heures et une fois après le démarrage.

Cet outil peut être assez utile s'il est personnalisé correctement pour alerter l'administrateur d'événements système inhabituels. **logcheck** peut être complètement personnalisé pour envoyer des courriers selon les événements récupérés des journaux et qui sont dignes d'attention. L'installation par défaut inclut des profils pour des événements ignorés et des violations de règles pour trois configurations différentes (station de travail, serveur et paranoïaque). Le paquet Debian contient un fichier de configuration `/etc/logcheck/logcheck.conf`, qui définit à quel utilisateur sont envoyés les vérifications. Il permet également aux paquets qui fournissent des services d'implémenter de nouvelles règles dans les répertoires: `/etc/logcheck/cracking.d/_paquet_`, `/etc/logcheck/violations.d/_paquet_`, `/etc/logcheck/violations.ignore.d/_paquet_`, `/etc/logcheck/ignore.d.paranoid/_paquet_`, `/etc/logcheck/ignore.d.server/_paquet_` et `/etc/logcheck/ignore.d.workstation/_paquet_`. Cependant, peu de paquets le font actuellement. Si vous avez une règle qui peut être utile à d'autres utilisateurs, veuillez l'envoyer comme un rapport de bogue sur le paquet approprié (comme un bogue de gravité *wishlist*). Pour obtenir plus de renseignements, veuillez consulter `/usr/share/doc/logcheck/README.Debian`.

Le meilleur moyen de configurer **logcheck** est d'éditer son fichier de configuration principal `/etc/logcheck/logcheck.conf` après l'avoir installé. Modifiez l'utilisateur par défaut (root) à qui seront envoyés par courrier les comptes-rendus. Vous devriez également y positionner le niveau de compte-rendu. logcheck-database a trois niveaux de compte-rendu de verbosité croissante: station de travail, serveur, paranoïaque. «serveur» étant le niveau par défaut, «paranoïaque» n'est recommandé que pour les machines de haute sécurité ne faisant fonctionner qu'aussi peu de services que possible et «station de travail» est pour les machines relativement protégés et non critiques. Si vous désirez ajouter de nouveaux fichiers journaux, ajoutez-les simplement à `/etc/logcheck/logcheck.logfiles`. Celui-ci est configuré pour une installation de syslog par défaut.

Une fois fait, vous pouvez vouloir vérifier les courriers envoyés, pour les quelques premiers jours, semaines ou mois. Si estimez recevoir des messages indésirables, ajoutez simplement l'expression rationnelle (consultez regex(7) et egrep(1)) qui correspond à ces messages au fichier `/etc/logcheck/ignore.d.niveau_de_compte-rendu /local`. Essayez de faire correspondre à la ligne entière du journal. Des détails sur l'écriture des règles sont expliqués dans `/usr/share/doc/logcheck-database/README.logcheck-database.gz`. C'est un processus d'affinement perpétuel; une fois que les messages envoyés sont toujours pertinents, vous pouvez considérer que l'affinement est terminé. Notez que si **logcheck** ne trouve rien de pertinent dans le système, il ne vous enverra

pas de courrier même s'il fonctionne (donc, vous pouvez ne recevoir de courrier qu'une fois par semaine si vous êtes chanceux).

Configurer l'endroit où les alertes sont envoyées

Debian livre une configuration standard de syslog (dans `/etc/syslog.conf`) qui archive les messages dans les fichiers appropriés en fonction de la facilité du système. Vous devriez être familier avec cela; jetez un œil au fichier `syslog.conf` et à la documentation si vous ne l'êtes pas. Si vous avez l'intention de maintenir un système sécurisé, vous devriez être conscient de l'endroit où les journaux sont envoyées ainsi ils ne sont pas perdus dans la nature.

Par exemple, envoyer des messages à la console est également utile pour de nombreux systèmes de production. Mais pour de nombreux systèmes semblables il est également important d'ajouter une nouvelle machine qui servira de serveur de journalisation (il reçoit les journaux de tous les autres systèmes).

Le courrier du superutilisateur devrait être pris en considération également, de nombreux contrôles de sécurité (comme snort) envoient des alertes dans la boîte aux lettres du superutilisateur. Celle-ci pointe généralement sur le premier utilisateur créé sur le système (vérifiez `/etc/aliases`). Veillez à envoyer le courrier du superutilisateur à un endroit où il sera lu (soit localement soit à distance).

Il y a d'autres comptes et alias «rôles» sur le système. Sur un petit système, le plus simple est probablement de s'assurer que tous ces alias pointent vers le compte du superutilisateur, et que les messages à destination du superutilisateur sont retransmis vers la boîte aux lettres personnelle de l'administrateur système.

FIXME : Il serait intéressant de dire comment un système Debian peut envoyer/recevoir des messages SNMP relatifs à des problèmes de sécurité (jfs). Voir: snmptragfmt, snmp et snmpd.

Utilisation d'un hôte d'archivage (loghost)

Un loghost est un hôte qui recueille les données des syslog à travers le réseau. Si l'une de vos machines est piratée, l'intrus n'est pas capable de dissimuler ses traces, à moins qu'il ne pirate également le loghost. Par conséquent, le loghost devrait être particulièrement sécurisé. Faire d'une machine un loghost est simple. Il suffit juste de démarrer le **syslogd** avec:

```
syslogd -r
```

et un nouveau loghost est né. De façon à rendre cela permanent dans Debian, éditez `/etc/default/syslogd` et changez la ligne

```
SYSLOGD=""
```

par

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

Ensuite, configurez les autres machines afin qu'elles envoient les données au loghost. Ajoutez une entrée comme celle qui suit dans `/etc/syslog.conf`:

```
  facilité.niveau           @votre_loghost
```

Consultez la documentation pour savoir ce qu'on peut utiliser à la place de *facilité* et *niveau* (ils ne devraient pas être mot pour mot comme cela). Si vous voulez tout archiver à distance, il suffit d'écrire:

```
*.*                     @votre_loghost
```

dans `syslog.conf`. Archiver à distance ainsi que localement est la meilleure solution (le pirate peut estimer avoir couvert ses traces après la suppression des fichiers de journalisation locaux). Consultez les pages de manuel syslog(3), syslogd(8) and syslog.conf(5) pour toutes informations complémentaires.

Permissions du fichier de journalisation

Il est important de décider non seulement comment les alertes sont utilisées, mais aussi qui y accède, c'est-à-dire qui peut lire ou modifier les fichiers de journalisation (en absence d'hôte d'archivage). Les alertes de sécurité que l'attaquant peut modifier ou désactiver sont de peu de valeur en cas d'intrusion. Vous devez également prendre en compte que les fichiers de journalisation peuvent révéler un grand nombre d'informations à propos du système à un intrus s'il y a accès.

Certaines permissions de fichiers de journalisation ne sont pas parfaites après l'installation (mais, bien sûr, cela dépend vraiment de vos règles de sécurité locales). Premièrement `/var/log/lastlog` et `/var/log/faillog` n'ont pas besoin d'être lisibles par les utilisateurs normaux. Dans `lastlog`, vous pouvez voir qui s'est connecté récemment, et dans `faillog`, vous voyez un résumé des connexions qui ont échouées. L'auteur recommande de faire un **chmod 660** sur les deux fichiers. Faites un tour rapide des fichiers de journalisation et décidez avec beaucoup d'attention quels fichiers de journalisation vous rendez lisible ou modifiable par un utilisateur avec un UID différent de 0 et un autre groupe que «adm» ou «root». Vous pouvez facilement vérifier cela sur le système avec:

```
#  find /var/log -type f -exec ls -l {} \; | cut -c 17-35 |sort -u
(voir à quels utilisateurs appartiennent les fichiers de /var/log)
#  find /var/log -type f -exec ls -l {} \; | cut -c 26-34 |sort -u
(voir à quels groups appartiennent les fichiers de /var/log)
# find /var/log -perm +004
(fichiers lisibles par tout utilisateur)
#  find /var/log \! -group root \! -group adm -exec ls -ld {} \;
(fichiers appartenant à des groupes autres que root ou adm)
```

Pour personnaliser la façon dont les fichiers de journalisation sont créés, vous devez probablement personnaliser le programme qui les génère. Cependant, si le fichier de journalisation est archivé, vous pouvez personnaliser le comportement de la création et de l'archivage.

Les utilitaires pour ajouter des correctifs au noyau

Debian GNU/Linux fournit quelques correctifs pour le noyau Linux qui améliorent sa sécurité du système. En voici quelques-uns.

- LIDS — http://www.lids.org fourni dans le paquet kernel-patch-2.4-lids. Ce correctif du noyau rend le processus de renforcement d'un système Linux plus facile en vous permettant de restreindre, cacher et protéger des processus, même par rapport au superutilisateur. Elle implémente des fonctionnalités de contrôle d'accès obligatoire («Mandatory Access Control»).

- http://trustees.sourceforge.net/ fourni dans le paquet trustees. Ce correctif ajoute un système avancé décent de gestion des permissions au noyau Linux. Des objets spéciaux (les « trustees ») sont associés à chaque fichier ou répertoire et ils sont stockés dans la mémoire noyau, ce qui permet un accès rapide pour toutes les 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.

- Le http://people.redhat.com/mingo/exec-shield/ fourni dans le paquet kernel-patch-exec-shield. Ce correctif fournit une protection contre plusieurs dépassements de tampon (attaques par écrasement de pile).

- The Grsecurity patch [http://www.grsecurity.net/], provided by the kernel-patch-2.4-grsecurity and kernel-patch-grsecurity2 packages [28] 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].

- Le kernel-patch-adamantix fournit les correctifs développés pour http://www.adamantix.org/, une distribution basée sur Debian. Le correctif noyau pour les versions 2.4.x du noyau introduit des fonctionnalités de sécurité comme une pile non exécutable grâce à l'utilisation de http://pageexec.virtualave.net/ et du contrôle d'accès obligatoire basé sur http://www.rsbac.org/. Parmi les autres fonctionnalités, on trouve: http://www.vanheusden.com/Linux/sp/, le périphérique boucle chiffré AES, la gestion MPPE et un rétroportage de la version 2.6 d'IPsec.

- cryptoloop-source. Ce correctif vous permet d'utiliser les fonctions de l'API de chiffrement du noyau pour créer des systèmes de fichiers chiffrés en utilisant le périphérique «loopback».

- Prise en charge d'IPsec par le noyau (du paquet linux-patch-openswan). Si vous voulez utiliser le protocole IPsec avec Linux, vous avez besoin de ce correctif. Vous pouvez ainsi créer des VPN très facilement, même vers les machines Windows, puisque IPsec est une norme courante. Des fonctionnalités IPsec ont été ajoutées au noyau de développement 2.5, cette fonctionnalité sera donc présente par défaut dans le futur noyau Linux 2.6. Site Internet: http://www.openswan.org. *FIXME*: les derniers noyaux2.4 fournis dans Debian incluent un rétroportage du code IPsec du noyau2.5. Commentaire sur cela.

Les correctifs de sécurité du noyau suivants ne sont disponibles que pour d'anciennes versions du noyau dans *Woody* et ils sont obsolètes:

- http://acl.bestbits.at/ (ACL) pour Linux fourni dans le paquet kernel-patch-acl. Ce correctif du noyau ajoute les listes de contrôle d'accès, une méthode avancée pour restreindre l'accès aux fichiers, par le noyau Linux. Cela vous permet de contrôler finement l'accès aux fichiers et répertoires.

- http://www.openwall.com/linux/ par Solar Designer, fourni dans le paquet kernel-patch-2.2.18-openwall. C'est un ensemble utile de restrictions pour le noyau, comme la restriction de liens, FIFO dans `/tmp`, une restriction de `/proc`, une gestion de descripteur de fichiers spéciaux, une pile de l'utilisateur non exécutable et bien plus. Note: ce paquet s'applique à la version2.2, aucun paquet n'est disponible pour les correctifs de la version2.4 fournie par Solar.

- kernel-patch-int. Ce correctif vous permet également d'ajouter des fonctionnalités de cryptographie au noyau Linux et était utile pour les versions de Debian jusqu'à Potato. Il ne fonctionne pas avec Woody

[28] 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

et si vous utilisez Sarge ou une version plus récente, vous devriez utiliser un noyau plus récent qui inclut déjà ces fonctionnalités.

Cependant, certains correctifs ne sont pas encore fournis dans Debian. Si vous croyez que certains devraient être inclus, veuillez le demander sur la page des http://www.debian.org/devel/wnpp.

Se protéger contre les dépassements de tampon

Dépassement de tampon est le nom d'une attaque courante sur un logiciel[29] qui utilise insuffisamment des vérifications de limites (une erreur de programmation courante, plus particulièrement en langage C) pour exécuter du code machine par des entrées de programme. Ces attaques, contre des logiciels serveur qui attendent des connexions distantes et contre des logiciels locaux qui autorisent des droits élevés aux utilisateurs (`setuid` ou `setgid`) peuvent avoir pour conséquence la compromission de tout un système.

Quatre méthodes en particulier permettent de se protéger contre les dépassement de tampon:

- appliquer un correctif au noyau pour empêcher l'exécution de la pile. Vous pouvez utiliser Exec-shield, OpenWall ou PaX (incluant les correctifs Grsecurity et Adamantix);
- corriger le code source en utilisant des outils pour trouver des fragments qui pourraient introduire cette faille;
- recompiler le code pour introduire des vérifications qui empêchent les dépassements en utilisant le correctif pour GCC http://www.research.ibm.com/trl/projects/security/ssp/ (qui est utilisé par http://www.adamantix.org).

Debian GNU/Linux, dans sa version3.0, fournit des logiciels pour implémenter toutes ces méthodes à l'exception de la protection de la compilation du code source (mais cela a été demandé dans le http://bugs.debian.org/213994).

Notez que même si Debian fournissait un compilateur qui fournit cette fonction de protection de dépassement de tampon/pile, tous les paquets auraient besoin d'être recompilés pour introduire cette fonctionnalité. C'est, en fait, ce que fait Adamantix (entre autres fonctionnalités). L'effet de cette nouvelle fonctionnalité sur la stabilité des logiciels doit encore être déterminée (certains programmes ou architectures de processeur pourraient être cassés à cause d'elle).

Dans tous les cas, soyez conscient que même ces contournement peuvent ne pas prévenir les dépassements de tampon car il existe des moyens de circonvenir ceux-ci, comme décrit dans l'http://packetstorm.linuxsecurity.com/mag/phrack/phrack58.tar.gz du magazine phrack ou dans l'alerte du CORE http://online.securityfocus.com/archive/1/269246.

Si vous voulez tester la protection contre les dépassements de tampon une fois que vous l'avez mise en place (quelque que soit la méthode), vous pouvez vouloir installer le paxtest et exécuter les tests qu'il fournit.

Correctif du noyau de protection pour les dépassements de tampon

Des correctifs du noyau liés aux dépassements de tampon incluant le correctif Openwall fournissent une protection contre les dépassements de tampon dans les noyaux Linux2.2. Pour les noyaux2.4 et plus ré-

[29] Si commune, en fait, qu'elles ont été la base de 20% des failles de sécurité signalés cette année, d'après les http://icat.nist.gov/icat.cfm?function=statistics.

cents, vous devrez utiliser l'implémentation Exec-shield ou l'implémentation PaX (fournies dans le correctif grsecurity, kernel-patch-2.4-grsecurity et dans le correctif Adamantix, kernel-patch-adamantix). Pour plus d'informations sur l'utilisation de ces correctifs, veuillez consulter la section la section intitulée « Les utilitaires pour ajouter des correctifs au noyau ».

Tester des programmes pour les dépassements

L'utilisation d'outils pour détecter des dépassements de tampon nécessitent dans tous les cas une expérience de programmation pour corriger (et recompiler) le code. Debian fournit par exemple: bfbtester (un testeur de dépassement de tampon qui brutalise des binaires par la force par des dépassements de ligne de commande et d'environnement). D'autres paquets intéressants pourraient aussi être rats, pscan, flawfinder et splint.

Sécurisation des transferts de fichiers

Pendant l'administration normale du système, il est habituellement nécessaire de transférer des fichiers à partir et vers le système installé. La copie des fichiers de façon sécurisée d'un hôte vers un autre peut être effectuée en utilisant le paquet serveur ssh. Une autre possibilité est d'utiliser ftpd-ssl, un serveur FTP qui utilise *Secure Socket Layer* pour chiffrer les transmissions.

Toutes ces méthodes nécessitent des clients spécifiques. Debian fournit des clients logiciels, comme **scp** du paquet ssh, qui fonctionne comme **rcp**, mais est complètement chiffré, donc les *méchants* ne peuvent même pas savoir CE QUE vous copiez. Il existe également un paquet client ftp-ssl pour le serveur équivalent. Vous pouvez trouver des clients pour ces logiciels, même pour d'autres systèmes d'exploitation (non UNIX), **putty** et **winscp** fournissent des implémentations de copie sécurisée pour toutes les versions des systèmes d'exploitation de Microsoft.

Notez qu'utiliser **scp** fournit un accès pour tous les utilisateurs à tout le système de fichiers à moins de faire un **chroot** comme décrit dans la section intitulée « Chrooter SSH ». L'accès FTP peut être **chroot**é, c'est probablement plus facile selon le démon que vous choisissez, comme décrit dans la section intitulée « Sécurisation de FTP ». Si vous vous inquiétez d'utilisateurs locaux pouvant parcourir les fichiers locaux et que vous voulez avoir une communication chiffrée, vous pouvez soit utiliser un démon FTP avec la prise en charge SSL, soit combiner un FTP sans chiffrement avec une configuration VPN (consultez la section intitulée « Réseaux Privés Virtuels »).

Limites et contrôle des systèmes de fichiers

Utilisation de quotas

Avoir une bonne politique relative aux quotas est important, vu qu'elle empêche les utilisateurs de remplir les disques durs.

Vous pouvez utiliser deux systèmes de quotas différents: les quotas utilisateur et les quotas groupe. Comme vous l'avez probablement deviné, les quotas utilisateur limitent la quantité d'espace qu'un utilisateur peut avoir, les quotas groupe quant à eux font la même chose pour les groupes. Retenez cela quand vous calculerez les tailles des quotas.

Il y a quelques points importants auxquels il faut penser dans la mise en place d'un système de quotas:

- garder les quotas suffisamment petits, ainsi les utilisateurs ne dévoreront pas l'espace disque ;
- garder les quotas suffisamment grands, ainsi les utilisateurs ne se plaindront pas et leur quota de courrier leur permettra d'accepter des courriers pendant une longue période ;

- utiliser des quotas sur tous les espaces accessibles en écriture par les utilisateurs, aussi bien sur `/home` que `/tmp`.

Tous les répertoires et partitions auxquels les utilisateurs ont accès en écriture complet devraient avoir les quotas activés. Recherchez ces partitions et répertoires et calculez une taille adaptée qui combine disponibilité et sécurité.

Bon, maintenant vous désirez utiliser les quotas. Avant tout, vous avez besoin de vérifier si vous avez activé la prise en charge des quotas dans le noyau. Si non, vous devrez le recompiler. Après cela, contrôlez si le paquet **quota** est installé. Si non, vous en aurez également besoin.

L'activation des quotas pour des systèmes de fichiers différents est aussi facile que la modification du paramètre `defaults` en `defaults,usrquota` dans le fichier `/etc/fstab`. Si vous avez besoin des quotas par groupe, remplacez `usrquota` par `grpquota`. Vous pouvez également utiliser les deux. Ensuite, créez des fichiers vides quota.user et quota.group à la racine du système de fichiers sur lequel vous voulez utiliser les quotas:

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

pour un système de fichiers `/home`)

Redémarrez **quota** en faisant:

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

Maintenant les quotas devraient être en fonction et leurs tailles peuvent être configurées.

L'édition de quotas pour un utilisateur spécifique peut être réalisée en faisant

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

. Les quotas par groupes peuvent être modifiés avec

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

. Ensuite, paramétrez les quotas soft et hard ou les quotas pour inœuds selon vos besoins.

Pour plus d'informations concernant les quotas, consultez la page de manuel de la commande quota et le quota mini-howto (`/usr/share/doc/HOWTO/fr-html/Quota.html`). Vous pouvez également vouloir étudier `pam_limits.so`.

Les attributs spécifiques du système de fichiers ext2 (chattr/lsattr)

En plus des permissions standards UNIX, les systèmes de fichiers ext2 et ext3 offrent un ensemble d'attributs spécifiques qui donnent plus de contrôle sur les fichiers du système. À la différence des permissions de base, ces attributs ne sont pas affichés par la commande standard **ls -l**, ni changés par la commande **chmod** et vous avez besoin de deux autres utilitaires, **lsattr** et **chattr** (du paquet e2fsprogs) pour les gérer. Notez que cela veut dire que ces attributs ne sont habituellement pas enregistrés quand vous sauvegardez le système, donc si vous modifiez l'un d'entre eux, il peut être utile d'enregistrer les commandes **chattr** successives dans un script pour pouvoir les repositionner plus tard si vous avez à récupérer une sauvegarde.

Parmi tous les attributs disponibles, les deux plus importants pour améliorer la sécurité sont référencés par les lettres «i» et «a» et ils ne peuvent être positionnés (ou enlevés) que par le superutilisateur:

- l'attribut «i» (inchangeable, «immutable»): un fichier ayant cet attribut ne peut-être ni modifié ni effacé ou encore renommé et aucun lien ne peut le référencer, même par le superutilisateur ;

- l'attribut «a» (ajout, «append»): cet attribut a le même effet que l'attribut «immutable», excepté que vous pouvez encore ouvrir le fichier en mode ajout. Cela veut dire que vous pouvez encore ajouter plus de contenu au fichier, mais qu'il est impossible de modifier un contenu précédent. Cet attribut est particulièrement utile pour les fichiers de journalisation stockés dans `/var/log/`, bien que vous devez considérer qu'ils sont parfois déplacés à cause des scripts d'archivage.

Ces attributs peuvent également être positionnés pour les répertoires, dans ce cas, le droit de modifier le contenu de la liste d'un répertoire est refusé (par exemple, renommer ou supprimer un fichier, etc.) Quand il est appliqué à un répertoire, l'attribut d'ajout ne permet que la création de fichiers.

Il est aisé de voir que l'attribut «a» améliore la sécurité, en donnant aux programmes qui ne sont pas exécutés par le superutilisateur, la possibilité d'ajouter des données à un fichier sans pouvoir modifier son précédent contenu. D'un autre côté, l'attribut «i» semble moins intéressant: après tout, le superutilisateur peut déjà utiliser les permissions standards UNIX pour restreindre l'accès à un fichier et un intrus qui aurait accès au compte superutilisateur peut toujours utiliser le programme **chattr** pour supprimer l'attribut. Un tel intrus peut tout d'abord être perplexe quand il se rendra compte qu'il ne peut pas supprimer un fichier, mais vous ne devriez pas supposer qu'il est aveugle — après tout, il est entré dans le système! Certains manuels (y compris une précédente version de ce document) suggèrent de supprimer simplement les programmes **chattr** et **lsattr** du système pour améliorer la sécurité, mais ce genre de stratégie, aussi connu comme «sécurité par l'obscurité», doit être absolument évitée, car elle donne un sentiment trompeur de sécurité.

Une façon sûre de résoudre ce problème est d'utiliser les fonctionnalités du noyau Linux, comme décrit dans la section intitulée « Défense proactive ». La fonctionnalité intéressante est appelée ici `CAP_LINUX_IMMUTABLE`: si vous la supprimez de l'ensemble des fonctionnalités (en utilisant par exemple la commande **lcap `CAP_LINUX_IMMUTABLE`**), il ne sera plus possible de modifier les attributs « a » ou « i » sur le système, même pour le superutilisateur ! Une stratégie complète serait alors la suivante :

- positionner les attributs «a» et «i» sur tous les fichiers voulus;

- ajouter la commande **lcap `CAP_LINUX_IMMUTABLE`** (ainsi que **lcap `CAP_SYS_MODULE`**, comme suggéré dans la section intitulée « Défense proactive ») à l'un des scripts de démarrage ;

- positionner l'attribut «i» sur ce script et les autres fichiers de démarrage, ainsi que sur le binaire **lcap** lui-même;

- exécuter la commande ci-dessus vous-même (ou réamorcer le système pour vous assurer que tout fonctionne comme prévu).

Maintenant que la fonctionnalité a été enlevée du système, un intrus ne peut plus changer aucun attribut des fichiers protégés et donc, il ne peut pas changer ou supprimer les fichiers. S'il force la machine à redémarrer (ce qui est la seule façon de récupérer le jeu de fonctionnalités), cela sera facile à détecter et la fonctionnalité sera de toute façon enlevée à nouveau dès que le redémarrage du système. La seule façon de changer un fichier protégé serait de réamorcer le système en mode utilisateur seul (single-user mode) ou d'utiliser une autre image d'amorçage, deux opérations qui nécessitent un accès physique à la machine!

Vérifier l'intégrité des systèmes de fichiers

Êtes-vous sûr que le /bin/login présent sur le disque dur est le même que celui que vous aviez installé il y a de cela quelques mois? Que faire si c'est une version piratée, qui enregistre les mots de passe entrés dans un fichier caché ou les envoie en clair sur Internet?

La seule méthode pour avoir un semblant de protection est de vérifier vos fichiers tous les heures/jours/mois (je préfère quotidiennement) en comparant l'actuel et l'ancien md5sum de ce fichier. Deux fichiers ne

peuvent avoir le même md5sum (le MD5 est basé sur 128 bits, ainsi la chance que deux fichiers différents aient le même md5sum est approximativement de un sur 3.4e3803), donc de ce côté tout est bon, à moins que quelqu'un ait piraté également l'algorithme qui crée les md5sums sur cette machine. C'est extrêmement difficile et très improbable. Vous devriez vraiment prendre en compte que la vérification de vos binaires est très importante étant donné que c'est un moyen facile de reconnaître des changements sur vos binaires.

Les outils couramment utilisés pour cela sont sxid, aide (Advanced Intrusion Detection Environment), tripwire, integrit et samhain. Installer `debsums` vous aidera également à vérifier l'intégrité du système de fichiers en comparant le md5sum de chaque fichier avec celui utilisé dans l'archive des paquets Debian. Mais faites attention: ces fichiers peuvent facilement être modifiés par un attaquant et tous les paquets ne fournissent pas de listes de md5sum pour les binaires qu'ils fournissent. Pour plus d'informations, veuillez consulter la section intitulée « Tests d'intégrité périodiques » et la section intitulée « Prendre un instantané («snapshot») du système ».

Vous pouvez vouloir utiliser **locate** pour indexer le système de fichiers en entier; si vous faites cela, envisagez les implications de cette action. Le paquet findutils de Debian contient **locate** qui s'exécute en tant qu'utilisateur nobody, ainsi, il indexe les fichiers qui sont visibles à tous les utilisateurs. Cependant, si vous changez son comportement, vous rendrez les emplacements de tous les fichiers visibles à tous les utilisateurs. Si vous voulez indexer tout le système de fichiers (pas les parties que l'utilisateur nobody peut voir), vous pouvez remplacer **locate** par slocate. slocate est étiqueté comme une version améliorée au niveau sécurité de GNU locate, mais il fournit en fait une fonctionnalité de localisation de fichier supplémentaire. Quand il utilise **slocate**, l'utilisateur ne peut voir que les fichiers auxquels il a vraiment accès et vous pouvez exclure tout fichier ou répertoire du système. Le paquet slocate exécute le processus de mise à jour avec des privilèges augmentés par rapport à locate et il indexe tous les fichiers. Les utilisateurs peuvent alors rechercher rapidement tout fichier qu'ils peuvent voir. **slocate** ne leur laisse pas voir les nouveaux fichiers; il filtre la sortie selon l'UID.

Vous pourriez utiliser bsign ou elfsign. elfsign fournit un utilitaire pour ajouter une signature numérique à un binaire ELF et un autre pour vérifier cette signature. L'actuelle implémentation utilise PKI pour signer la somme de contrôle du binaire. L'avantage de faire cela est que ceux qui le veulent peuvent déterminer si un binaire a été modifié et qui l'a créé. bsign utilise GPG, elfsign utilise les certificats PKI (X.509, OpenSSL).

Mise en place de la vérification setuid

Le paquet Debian checksecurity fournit une tâche **cron** qui s'exécute tous les jours dans `/etc/cron.daily/checksecurity` [30]. Cette tâche **cron** exécutera le script **/usr/sbin/checksecurity** qui sauvegardera les renseignements sur les modifications.

Le comportement par défaut est de ne pas envoyer cette information au superutilisateur mais à la place de garder une copie quotidienne des modifications dans `/var/log/setuid.changes`. Vous devrez positionner la variable MAILTO (dans `/etc/checksecurity.conf`) à «root» pour que ces renseignements lui soient envoyés. Consultez checksecurity(8) pour plus d'informations sur la configuration.

Sécurisation des accès réseau

FIXME : Besoin de plus de contenu (spécifique à Debian).

Configuration des options réseau du noyau

Beaucoup de fonctionnalités du noyau peuvent être modifiées en cours de fonctionnement en envoyant quelque chose (par la commande **echo**) dans le système de fichiers `/proc` ou en utilisant **/sbin/sysctl**. En

[30] Dans les versions précédentes, checksecurity était intégré dans cron et le fichier était `/etc/cron.daily/standard`.

entrant **/sbin/sysctl** `-A`, vous pouvez voir ce que vous pouvez configurer et quelles sont les options, elles peuvent être modifiées en exécutant:

```
/sbin/sysctl -w variable=valeur
```

(consultez sysctl(8)). Vous aurez seulement en de rares occasions à éditer quelque chose ici, mais vous pouvez augmenter la sécurité de cette manière aussi. Par exemple :

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

C'est un «émulateur Windows» parce que ça agit comme Windows sur les ping de broadcast si celui-ci est positionné à 1. C'est-à-dire que les requêtes d'echo ICMP envoyées à l'adresse broadcast seront ignorées. Sinon, cela ne fait rien.

Si vous voulez empêcher le système de répondre aux requêtes d'echo ICMP, activez cette option de configuration:

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

Pour enregistrer les paquets avec des adresses impossibles (à cause de routes erronées) sur le réseau, utilisez:

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

Pour plus d'informations sur ce qui peut être fait avec `/proc/sys/net/ipv4/*`, consultez `/usr/src/linux/Documentation/filesystems/proc.txt`. Toutes les options sont décrites de façon complète sous `/usr/src/linux/Documentation/networking/ip-sysctl.txt` [31].

Configurer syncookies

Cette option est à double tranchant. D'un côté, elle protège le système contre le syn packet flooding; d'un autre côté, elle viole les standards définis (RFCs).

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

Si vous voulez changer cette option à chaque fois que le noyau fonctionne, vous devez le faire dans `/etc/network/options` en positionnant `syncookies=yes`. Cela prendra effet à chaque fois que `/etc/init.d/networking` est exécuté (ce qui est habituellement fait lors du démarrage) tandis que la commande suivante aura un effet unique jusqu'au prochain redémarrage:

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

Cette option n'est dispobile que si vous avez compilé le noyau avec `CONFIG_SYNCOOKIES`. Tous les noyaux Debian sont compilés avec cette option incluse, mais vous pouvez le vérifier en exécutant:

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

[31] Dans Debian, les paquets kernel-source-`version` copient les sources sous `/usr/src/kernel-source-version.tar.bz2`, remplacez simplement `version` par la version des sources du noyau installé.

Pour plus d'informations sur les syncookies TCP, consultez http://cr.yp.to/syncookies.html.

Sécurisation du réseau pendant l'amorçage

Quand vous positionnez des options de configuration de réseau du noyau, vous devez le configurer pour que ce soit chargé à chaque fois que le système est redémarré. L'exemple suivant active un grand nombre des options précédentes ainsi que d'autres options utiles.

Il y a en fait deux façons de configurer le réseau au démarrage. Vous pouvez configurer `/etc/sysctl.conf` (consultez sysctl.conf(5)) ou introduire un script qui est appelé quand l'interface est activée. La première option sera appliquée à toutes les interfaces alors que la seconde option vous permettra de configurer cela interface par interface.

Un exemple de fichier de configuration `/etc/sysctl.conf` qui sécurisera quelques options de réseau au niveau du noyau est présenté ci-dessous. Notez les commentaires dans ce fichier, `/etc/network/options` peut forcer certaines options si elles sont en contradiction avec celles de ce fichier lors de l'exécution de `/etc/init.d/networking` (ce qui a lieu après `procps` dans la séquence de démarrage).

```
#
# /etc/sysctl.conf - Fichier de configuration pour positionner les
# variables système
# Consultez sysctl.conf(5) pour plus de renseignements. Consultez
# également les fichiers sous Documentation/sysctl/,
# Documentation/filesystems/proc.txt et
# Documentation/networking/ip-sysctl.txt dans les sources du noyau
# (/usr/src/kernel-$version si vous avez installé un paquet de noyau)
# pour plus d'informations sur les valeurs qui peuvent être définies ici.

#
# Attention : /etc/init.d/procps est exécuté pour positionner les
# variables suivantes. Cependant, après cela, /etc/init.d/networking
# positionne certaines options réseau avec des valeurs intrinsèques. Ces
# valeurs peuvent être forcées en utilisant /etc/network/options.
#
#kernel.domainname = example.com

# Paramètres supplémentaires - adapté du script fourni
# par Dariusz Puchala (voir ci-dessous)
# Ignorer les broadcasts ICMP
net/ipv4/icmp_echo_ignore_broadcasts = 1
#
# Ignorer les erreurs ICMP erronées
net/ipv4/icmp_ignore_bogus_error_responses = 1
#
# Ne pas accepter les redirections ICMP (empêche les attaques en
# homme au milieu)
net/ipv4/conf/all/accept_redirects = 0
# _ou_
# N'accepter les redirections ICMP que pour les passerelles
# de notre liste de passerelles par défaut (activé par défaut)
# net/ipv4/conf/all/secure_redirects = 1
#
```

```
# Ne pas accepter les redirections ICMP (ce n'est pas un routeur)
net/ipv4/conf/all/send_redirects = 0
#
# Ne pas faire suivre les paquets IP (ce n'est pas un routeur)
# Remarque : assurez-vous que /etc/network/options contient
# « ip_forward=no »
anet/ipv4/conf/all/forwarding = 0
#
# Activer les TCP Syn Cookies
# Remarque : assurez-vous que /etc/network/options contient
# « syncookies=yes »
net/ipv4/tcp_syncookies = 1
#
# Enregistrer les paquets martiens
net/ipv4/conf/all/log_martians = 1
#
# Activer la vérification d'adresse source pour toutes les
# interfaces pour empêcher certaines attaques par usurpation
# Remarque : assurez-vous que /etc/network/options contient
# « spoofprotect=yes »
net/ipv4/conf/all/rp_filter = 1
#
# Ne pas accepter les paquets de routage source IP
# (ce n'est pas un routeur)
net/ipv4/conf/all/accept_source_route = 0
```

Pour utiliser le script, vous devez tout d'abord le créer, par exemple, dans `/etc/network/interface-secure` (le nom est donné comme exemple) et l'appeler à partir de `/etc/network/interfaces` comme ceci:

```
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
```

Dans cet exemple, avant que l'interface eth0 ne soit activée, le script sera appelé pour sécuriser toutes les interfaces réseau comme montré ci-dessous.

```
#!/bin/sh -e
# Nom du script : /etc/network/interface-secure
#
# Modification de plusieurs comportements par défaut pour sécuriser contre
# certaines attaques et usurpations IP pour toutes les interfaces.
#
# Fourni par Dariusz Puchalak.
#

# Activation de la protection broadcast echo.
echo 1 > /proc/sys/net/ipv4/icmp_echo_ignore_broadcasts
```

```
# Désactivation de l'IP forwarding.
echo 0 > /proc/sys/net/ipv4/conf/all/forwarding

# Activation de la protection TCP syn cookies.
echo 1 > /proc/sys/net/ipv4/tcp_syncookies


# Enregistrement des paquets avec des adresses impossibles
# (cela comprend les paquets usurpés (spoofed), les paquets routés
# source, les paquets redirigés), mais faites attention à cela
# sur les serveurs web très chargés.
echo 1 >/proc/sys/net/ipv4/conf/all/log_martians

# Activation de la protection sur les mauvais messages d'erreur.
echo 1 > /proc/sys/net/ipv4/icmp_ignore_bogus_error_responses

# Protection d'usurpation IP.
echo 1 > /proc/sys/net/ipv4/conf/all/rp_filter

# Désactivation des redirections ICMP.
echo 0 > /proc/sys/net/ipv4/conf/all/accept_redirects
echo 0 > /proc/sys/net/ipv4/conf/all/send_redirects

# Désactivation des paquets source routés.
echo 0 > /proc/sys/net/ipv4/conf/all/accept_source_route

exit 0
```

Remarquez que vous pouvez en fait avoir des scripts par interface qui activeront différentes options réseau pour différentes interfaces (si vous en avez plus d'une), il vous suffit de changer la ligne pre-up en:

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

et utiliser un script qui n'applique les changements qu'à une interface spécifique et non à toutes les interfaces disponibles. Notez cependant que certaines options réseau ne peuvent être appliquées que globalement. Un exemple de script est celui-ci:

```
#!/bin/sh -e
# Nom du script : /etc/network/interface-secure
#
# Modifie plusieurs comportements par défaut pour sécuriser contre
# certaines attaques et usurpations TCP/IP pour une interface donnée.
#
# Fourni par Dariusz Puchalak.
#

IFACE=$1
if [ -z "$IFACE" ] ; then
   echo "$0 : un nom d'interface doit être fourni en argument"
   echo "Utilisation : $0 <interface>"
   exit 1
fi
```

```
if [ ! -e /proc/sys/net/ipv4/conf/$IFACE/ ]; then
   echo "$0 : l'interface $IFACE n'existe pas "
   echo "(impossible de trouver /proc/sys/net/ipv4/conf/)"
   exit 1
fi

# Désactivation de l'IP forwarding.
echo 0 > /proc/sys/net/ipv4/conf/$IFACE/forwarding

# Enregistrement des paquets avec des adresses impossibles
# (cela inclut les paquets usurpés (spoofed), les paquets routés
# source, les paquets redirigés), mais faites attention à cela
# sur les serveurs web très chargés.
echo 1 >/proc/sys/net/ipv4/conf/$IFACE/log_martians

# Protection d'usurpation IP.
echo 1 > /proc/sys/net/ipv4/conf/$IFACE/rp_filter

# Désactivation des redirections ICMP.
echo 0 > /proc/sys/net/ipv4/conf/$IFACE/accept_redirects
echo 0 > /proc/sys/net/ipv4/conf/$IFACE/send_redirects

# Désactivation des paquets source routés.
echo 0 > /proc/sys/net/ipv4/conf/$IFACE/accept_source_route

exit 0
```

Vous pouvez également créer un script `init.d` et le faire exécuter au démarrage (en utilisant **update-rc.d** pour créer les liens `rc.d` appropriés).

Configuration des fonctionnalités de pare-feu

De façon à avoir des privilèges de pare-feu, soit pour protéger le système local ou d'autres *derrière* lui, le noyau doit être compilé avec les options correspondant aux pare-feu. Le noyau standard de Debian 2.2 (Linux 2.2) fournit **ipchains** qui est un pare-feu pour filtrer les paquets, le noyau standard de Debian 3.0 (Linux 2.4) fournit lui le pare-feu **iptables** (netfilter).

Dans tous les cas, il est relativement facile d'utiliser un noyau différent de celui fourni par Debian. Vous pouvez trouver des noyaux précompilés sous forme de paquets que vous pouvez facilement installer sur le système Debian. Vous pouvez également télécharger les sources du noyau en utilisant kernel-source-X et construire des paquets de noyau personnalisé en utilisant **make-kpkg** du paquet kernel-package.

La mise en place de pare-feu dans Debian est abordée plus en détail dans la section intitulée « Ajouter des capacités au pare-feu ».

Désactiver les problèmes d'hôtes weak-end

Les systèmes avec plus d'une interface sur différents réseaux peuvent avoir des services configurés pour qu'ils ne puissent s'associer qu'à une adresse IP donnée. Cela prévient habituellement les accès aux services quand ils sont interrogés par une adresse donnée. Cependant, cela ne veut pas dire (bien qu'il s'agisse d'une erreur classique) que le service est lié à une adresse *matérielle* donnée (carte interface). [32]

[32] Pour reproduire cela (exemple fourni par Felix von Leitner sur la liste de diffusion Bugtraq):

Cela semble, cependant, ne pas fonctionner avec les services liés à 127.0.0.1, vous pourriez devoir écrire des tests utilisant des sockets bruts.

Ce n'est pas un problème ARP et ce n'est pas une violation de RFC (c'est ce que l'on appelle le *weak end host* dans la ftp://ftp.isi.edu/in-notes/rfc1122.txt, section 3.3.4.2). Rappelez-vous que les adresses IP n'ont rien à voir avec les interfaces physiques.

Sur les noyaux 2.2 (et antérieurs), cela peut être corrigé avec:

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

Sur les noyaux suivants, cela peut être corrigé avec:

- des règles iptables ;
- un routage correctement configuré[33] ;
- des correctifs du noyau[34].

Tout au long de ce texte, il y aura plusieurs occasions pour lesquelles il est affiché comment configurer certains services (serveur SSH, Apache, service d'impression, etc.) pour les avoir en attente sur une adresse donnée, le lecteur devra prendre en compte que, sans les correctifs donnés ici, le correctif n'empêchera pas les accès depuis le même réseau (local). [35]

FIXME : Commentaires sur Bugtraq indiquant qu'il existe une méthode spécifique à Linux pour associer à une interface donnée.

FIXME : Créer un bogue sur netbase pour que le correctif de routage soit le comportement standard dans Debian?

Protéger contre les attaques ARP

Quand vous ne faites pas confiance aux autres machines du réseau (ce qui devrait toujours être le cas parce que c'est l'attitude la plus sûre), vous devriez vous protéger contre les différentes attaques ARP existantes.

Comme vous le savez, le protocole ARP est utilisé pour lier des adresses IP à des adresses MAC (consultez la ftp://ftp.isi.edu/in-notes/rfc826.txt pour tous les détails). À chaque fois que vous envoyez un paquet à une adresse IP, une résolution ARP est effectuée (en regardant en premier dans le cache local ARP, puis si l'adresse IP n'est pas présente dans le cache, en diffusant une requête ARP) pour trouver l'adresse matérielle de la cible. Toutes les attaques ARP ont pour but d'amener la machine à croire que l'adresse IP de la machine B est associée à l'adresse MAC de la machine de l'intrus; puis tous les paquets que vous voudrez envoyer à l'adresse IP associée à la machine B seront envoyée à la machine de l'intrus, etc.

Ces attaques (empoisonnement du cache, falsification ARP, etc.) permettent à l'attaquant de renifler le trafic même sur des réseaux utilisant des switchs, pour pirater facilement des connexions, pour déconnecter

hôte a (eth0 connecté sur l'eth0 de l'hôte b) : ifconfig eth0 10.0.0.1 ifconfig eth1 23.0.0.1 tcpserver -RHl loc

[33] Le fait que ce comportement puisse être changé par le routage a été décrit par Matthew G. Marsh dans l'enfilade sur Bugtraq:

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.

[34] Il existe des correctifs disponibles pour ce comportement comme décrit dans l'enfilade sur Bugtraq à http://www.linuxvirtualserver.org/~julian/#hidden et http://www.fefe.de/linux-eth-forwarding.diff.

[35] Un attaquant peut avoir beaucoup de problèmes à transférer un accès après une configuration de l'adresse IP s'il n'est pas le domaine de broadcast (même réseau) que l'hôte attaqué. Si l'attaque passe par un routeur, il peut être assez difficile pour les réponses de retourner quelque part.

tout hôte du réseau, etc. Les attaques ARP sont puissantes et simples à implémenter et plusieurs outils existent comme **arpspoof** du paquet dsniff ou http://arpoison.sourceforge.net/.

Cependant, il existe toujours une solution:

- utiliser un cache ARP statique. Vous pouvez mettre en place des entrées «statiques» dans le cache ARP avec:

  ```
  arp -s nom_d_hôte adresse_matérielle
  ```

  En plaçant des entrées statiques pour chaque hôte important du réseau, vous garantissez que personne ne pourra créer ou modifier une entrée (dissimulée) pour ces hôtes (les entrées statiques n'expirent pas et elles ne peuvent pas être modifiées) et les réponses ARP falsifiées seront ignorées ;

- détecter le trafic ARP suspect. Vous pouvez utiliser arpwatch, karpski ou des IDS plus généraux qui peuvent également détecter le trafic ARP suspect (snort, http://www.prelude-ids.org, etc.) ;

- implémenter un filtrage de trafic IP validant l'adresse MAC.

Prendre un instantané («snapshot») du système

Avant de mettre le système en production, vous pouvez prendre un instantané du système entier. Cet instantané pourrait être utilisé en cas de compromission (consultez Chapitre 11, *Après la compromission (la réponse à l'incident)*). Vous devriez refaire cette mise à jour à chaque fois que le système est mis à jour, particulièrement si vous mettez à jour vers une nouvelle version de Debian.

Pour cela, vous pouvez utiliser un support inscriptible et amovible qui peut être positionné en lecture seule, ce peut être une disquette (en lecture seule après utilisation), un CD d'une unité de CD (vous pourriez utiliser un CD réinscriptible, ainsi vous pourriez même garder des sauvegardes des md5sums à différentes dates), ou un disque USB ou une carte MMC (si le système peut accéder à ceux-ci et qu'ils peuvent être protégés en écriture).

Le script suivant crée un tel instantané:

```
#!/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
```

Notez que le binaire md5sum (et le binaire sha1sum, s'il est disponible) est placé sur la disquette pour pouvoir être utilisé plus tard pour vérifier les binaires du système (juste au cas où il serait aussi corrompu). Cependant, si vous voulez vous assurer que vous exécutez bien un binaire légitime, vous pouvez vouloir, soit compiler une copie statique du binaire md5sum et utiliser celui-ci (pour empêcher une bibliothèque libc corrompue d'interférer avec le binaire), soit utiliser des instantanés de md5sums depuis un environnement propre exclusivement comme un CD de récupération ou un CD autonome (pour empêcher un noyau corrompu d'interférer). Je ne peux insister assez sur ce point: si vous êtes sur un système compromis, vous ne pouvez pas faire confiance à ce qui s'affiche, consultez Chapitre 11, *Après la compromission (la réponse à l'incident)*.

L'instantané n'inclut pas les fichiers sous `/var/lib/dpkg/info` qui incluent les sommes de hachage MD5 des paquets installés (dans les fichiers se terminant par `.md5sums`). Vous pourriez également y copier ces renseignements, veuillez cependant noter que:

- les fichiers md5sums incluent les md5sums de tous les fichiers fournis par les paquets Debian, pas seulement les binaires système. Par conséquent, la base de données est plus importante (5Mo contre 600ko dans un système Debian GNU/Linux avec un système graphique et environ 2,5Go de logiciels installés) et elle ne tiendra sur un petit support amovible (comme une simple disquette, mais tiendra sans doute sur une clef USB) ;
- tous les paquets Debian ne fournissent pas les md5sums pour les fichiers installé car ce n'est pas (actuellement) imposé par la Charte. Notez, cependant, que vous pouvez générer les md5sums pour tous les paquets en utilisant debsums après avoir fini l'installation du système:

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

Une fois que l'instantané est fait, vous devriez vous assurer de placer le support en lecture seule. Vous pouvez ensuite le stocker pour archivage ou le placer dans le lecteur et utiliser une vérification **cron** toutes les nuits en comparant les md5sums d'origine avec ceux de l'instantané.

Si vous ne voulez pas configurer de vérification manuelle, vous pouvez toujours utiliser n'importe quel système d'intégrité disponible qui fera cela et plus, pour de plus amples renseignements, veuillez consulter la section intitulée « Tests d'intégrité périodiques ».

Autres recommandations

N'utilisez pas de logiciels dépendant de svgalib

SVGAlib est très bien pour les amoureux de la console mais s'est montrée très peu sûre par le passé. Des exploitations de failles de **zgv** ont été diffusées et il était facile de devenir superutilisateur. Essayez d'éviter l'utilisation de programmes utilisant la SVGAlib chaque fois que c'est possible.

Chapitre 5. Sécurisation des services du système

Les services présents sur un système peuvent être sécurisés de deux façons :

- les rendre accessibles uniquement aux points d'accès (interfaces) nécessaires ;
- les configurer correctement ainsi seuls les utilisateurs habilités pourront les utiliser.

Restreindre les services pour qu'ils ne soient accessibles que depuis un endroit bien spécifique peut être fait au niveau du noyau (pare-feu), configurez les services pour écouter uniquement sur une interface définie (certains services ne fournissent peut-être pas cette fonctionnalité) ou utilisez tout autre méthode, par exemple le correctif vserver pour Linux (2.4.16) peut être utilisé pour forcer les processus à n'utiliser qu'une interface.

Concernant les services lancés par **inetd** (**telnet**, **ftp**, **finger**, **pop3**, etc.), il est à noter que inetd peut être configuré pour que les services n'écoutent que sur une interface précise (en utilisant la syntaxe `service@ip`), mais c'est une fonctionnalité non documentée. L'un de ses remplaçants, le métadémon **xinetd**, inclut une option `bind` pour faire cela. Consultez ixnetd.conf(5)

```
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
}
```

Les paragraphes suivants détaillent comment déterminer les services qui peuvent être configurés correctement en fonction de leur utilisation.

Sécurisation de SSH

Si vous utilisez toujours TELNET au lieu de SSH, vous devriez prendre une pause dans la lecture de ce manuel pour remédier à cela. SSH devrait être utilisé pour toutes les connexions distantes à la place de TELNET. À une époque où il est facile de scruter le trafic Internet et d'obtenir les mots de passe en clair, vous devriez utiliser uniquement les protocoles qui utilisent la cryptographie. Par conséquent, effectuez maintenant un `apt-get install ssh` sur le système.

Encourager tous les utilisateurs du système à utiliser SSH au lieu de TELNET, ou mieux encore, désinstallez telnet/telnetd. De plus, vous devriez éviter de vous connecter au système en utilisant SSH en tant que superutilisateur et préférer l'utilisation de méthodes alternatives pour devenir superutilisateur comme `su` ou `sudo`. Enfin, le fichier `sshd_config`, dans `/etc/ssh`, devrait être modifié comme suit pour accroître la sécurité.

- Ne faîtes écouter SSH que sur une interface donnée, juste au cas où vous en ayez plus d'une (et ne voulez pas que SSH y soit disponible) ou si vous ajoutez, dans le futur, une nouvelle carte réseau (et ne voulez pas de connexions SSH dessus).

- Essayez autant que possible de ne pas autoriser de connexion en tant que superutilisateur. Si quelqu'un veut devenir superutilisateur par SSH, deux connexions sont maintenant nécessaires et le mot de passe du superutilisateur ne peut être attaqué par force brute par SSH.

- `Port 666` ou `ListenAddress 192.168.0.1:666` change le port d'écoute, ainsi l'intrus ne peut être complètement sûr de l'exécution d'un démon sshd (soyez prévenus, c'est de la sécurité par l'obscurité).

- `PermitEmptyPasswords no` Les mots de passe vides sont un affront au système de sécurité.

- Autorise seulement certains utilisateurs à avoir accès par SSH à cette machine. `user@host` peut également être utilisé pour n'autoriser l'accès qu'à un utilisateur donné depuis un hôte donné.

- Autorise seulement certains membres de groupes à avoir accès par SSH à cette machine. AllowGroups et AllowUsers ont des directives équivalentes pour interdire l'accès à la machine. Sans surprise elles s'appellent « DenyUsers » et « DenyGroups ».

- Il vous appartient complètement de décider ce que vous voulez faire. Il est plus sûr d'autoriser l'accès à la machine uniquement aux utilisateurs avec des clefs SSH placées dans le fichier `~/.ssh/authorized_keys`. Si c'est ce que vous voulez, positionnez cette option à « no ».

- Désactiver toute forme d'autorisation dont vous n'avez pas réellement besoin si vous n'utilisez pas, par exemple, `RhostsRSAAuthentication`, `HostbasedAuthentication`, `KerberosAuthentication` ou `RhostsAuthentication`, vous devriez les désactiver même s'ils le sont déjà par défaut (consultez la page de manuel sshd_config(5)).

- Désactiver le protocole version 1, car il a des défauts de conception qui facilitent le piratage de mots de passe. Pour obtenir de plus amples renseignements, consultez http://earthops.net/ssh-timing.pdf ou le http://xforce.iss.net/static/6449.php.

- Ajouter une bannière (elle sera récupérée du fichier) pour les utilisateurs se connectant au serveur SSH. Dans certains pays, envoyer un avertissement avant l'accès à un système donné avertissant des accès non autorisés ou du suivi des utilisateurs devrait être ajouté pour avoir une protection légale.

Vous pouvez également restreindre l'accès au serveur ssh en utilisant `pam_listfile` ou `pam_wheel` dans le fichier de contrôle PAM. Par exemple, vous pourriez bloquer tous les utilisateurs qui ne sont pas dans `/etc/loginusers` en ajoutant cette ligne à `/etc/pam.d/ssh` :

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

Pour finir, soyez conscient que ces directives proviennent d'un fichier de configuration OpenSSH. Actuellement, trois démons SSH sont couramment utilisés, ssh1, ssh2, et OpenSSH par les gens d'OpenBSD. ssh1 était le premier démon SSH disponible et est toujours le plus couramment utilisé (il y a même des rumeurs à propos d'un portage pour Windows). ssh2 a beaucoup d'avantages par rapport à ssh1 sauf qu'il est diffusé sous une licence non libre. OpenSSH est un démon SSH complètement libre, qui gère à la fois ssh1 et ssh2. OpenSSH est la version installée sur Debian quand le paquet ssh est choisi.

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.

Chrooter SSH

OpenSSH ne fournit pas de moyen à l'heure actuelle pour chrooter automatiquement les utilisateurs lors de la connexion (la version commerciale fournit cette fonctionnalité). Cependant, il existe un projet ayant pour but de fournir cette fonctionnalité pour OpenSSH également, consultez http://chrootssh.sourceforge.net, il n'est cependant pas empaqueté pour Debian actuellement. Vous pourriez cependant utiliser le module `pam_chroot` module comme décrit dans la section intitulée « Restriction des utilisateurs ».

Dans la section intitulée « Environnement de chroot pour SSH », vous pouvez trouver plusieurs options pour créer un environnement chroot pour SSH.

Clients SSH

Si vous utilisez un client SSH pour se connecter au serveur SSH, vous devez vous assurer qu'il prend en charge les mêmes protocoles que ceux utilisés sur le serveur. Par exemple, si vous utilisez le paquet mindterm, il ne prend en charge que le protocole version 1. Cependant, le serveur sshd est, par défaut, configuré pour n'accepter que la version 2 (pour des raisons de sécurité).

Interdire les transferts de fichiers

Si vous ne voulez *pas* que les utilisateurs transfèrent des fichiers depuis et vers le serveur ssh, vous devez restreindre l'accès au **sftp-server** *et* l'accès **scp**. Vous pouvez restreindre **sftp-server** en configurant le bon `Subsystem` dans `/etc/ssh/sshd_config`.

Vous pouvez aussi cloisonner les utilisateurs dans un chroot (en utilisant libpam-chroot de telle sorte que même si le transfert de fichiers est autorisé, ils soient limités à un environnement qui ne contient aucun fichier système.

Restriction d'accès au seul transfert de fichiers

Vous pourriez restreindre l'accès aux utilisateurs pour leur permettre seulement le transfert de fichiers sans interpréteur de commandes interactif. Pour faire cela, vous pouvez :

- soit interdire les connexions d'utilisateurs au serveur SSH (comme décrit ci-dessus par le fichier de configuration ou par la configuration PAM) ;
- soit donner aux utilisateurs un interpréteur de commandes restreint comme scponly ou rssh. Ces interpréteurs de commandes restreignent les commandes disponibles pour les utilisateurs afin de ne pas leur donner de droits d'exécution à distance.

Sécurisation de Squid

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.

Voici ci-dessous la configuration minimum recommandée:

```
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  # ports non enregistrés
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
(...)
# Ne permet l'accès à cachemgr que depuis localhost
http_access allow manager localhost
http_access deny manager
# Ne permet des requêtes de purge que depuis localhost
http_access allow purge localhost
http_access deny purge
# Interdit les requêtes sur des ports inconnus
http_access deny !Safe_ports
# Interdit CONNECT sur tout autre port que SSL
http_access deny CONNECT !SSL_ports
#
# INSÉRER VOS PROPRES RÈGLES ICI POUR PERMETTRE L'ACCÈS
# DEPUIS LES CLIENTS
#
http_access allow localhost
# Et enfin, interdit tout autre accès à ce mandataire
http_access deny all
# Par défaut :
# icp_access deny all
#
# Permet les requêtes ICP à tout le monde
icp_access allow all
```

Vous pouvez également configurer Squid selon vos ressources système, en incluant la mémoire cache (option `cache_mem`), l'emplacement de vos fichiers du cache et la quantité d'espace qu'ils prendront sur disque (option `cache_dir`).

Notez que, s'il n'est pas configuré correctement, n'importe qui peut relayer un message par l'intermédiaire de Squid, puisque les protocoles HTTP et SMTP sont conçus de façon similaire. Le fichier de configuration par défaut interdit l'accès au port 25. Si vous voulez autoriser les connexions sur ce port, il vous faudra l'ajouter dans la liste des Safe_ports (ports autorisés). Cependant, ce n'est *PAS* recommandé.

Installer et configurer le serveur mandataire et le cache correctement ne représente qu'une partie de la sécurisation du site. Une autre tâche nécessaire réside dans l'analyse des journaux de Squid pour s'assurer que

tout fonctionne comme prévu. Quelques paquets dans Debian GNU/Linux peuvent aider l'administrateur dans cette tâche. Les paquets suivant sont disponibles dans Debian 3.0 et Debian 3.1 (Sarge) :

- calamaris - Analyseur des journaux pour serveurs mandataires Squid ou Oops
- modlogan - Analyseur modulaire de journaux
- sarg - Création de compte-rendu d'analyse de Squid
- squidtaild - Programme de surveillance des journaux de 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

Sécurisation de FTP

Si vous avez réellement besoin d'utiliser FTP (sans l'emballer avec sslwrap ou à l'intérieur d'un tunnel SSL ou SSH), vous devriez « chrooter » FTP dans le répertoire personnel de l'utilisateur, ainsi l'utilisateur ne pourra rien voir d'autre que ses propres répertoires. Autrement, il pourrait parcourir le système comme s'il disposait d'un interpréteur de commandes. Vous pouvez ajouter la ligne suivante dans la section global de `proftpd.conf` pour activer ce chroot :

```
DefaultRoot ~
```

Redémarrez ProFTPD par `/etc/init.d/proftpd restart` et vérifiez si vous pouvez sortir de votre propre répertoire personnel.

Pour prévenir ProFTPD d'attaques par déni de service avec l'utilisation de ../../.., ajoutez la ligne suivante dans `/etc/proftpd.conf` : `DenyFilter \*.*/`

Rappelez-vous toujours que FTP envoie les identifiants et les mots de passe d'authentification en clair (ce n'est pas un problème si vous fournissez un service public anonyme) et il existe de meilleures alternatives dans Debian pour cela. Par exemple, **sftp** (fourni par ssh). Il existe également d'autres implémentations de SSH pour d'autres systèmes d'exploitation : http://www.chiark.greenend.org.uk/~sgtatham/putty/ et http://www.cygwin.com par exemple.

Cependant, si vous maintenez encore le serveur FTP tout en donnant un accès par SSH aux utilisateurs, vous pouvez rencontrer un problème courant. Les utilisateurs accédant aux serveurs FTP anonymes à l'intérieur des systèmes sécurisés par SSH peuvent essayer de se connecter dans le *serveur FTP*. Bien que l'accès sera refusé, le mot de passe sera tout de même envoyé en clair sur le réseau. Pour éviter cela, le développeur de ProFTPD, TJ Saunders, a créé un correctif pour empêcher des utilisateurs de fournir au serveur FTP anonyme des comptes SSH valables. Plus d'informations et le correctif sont disponibles, consultez http://www.castaglia.org/proftpd/#Patches. Ce correctif a été également indiqué pour Debian, consultez le http://bugs.debian.org/145669.

Sécurisation de l'accès au système X Window

Actuellement, les terminaux X sont de plus en plus utilisés dans les entreprises où un seul serveur est nécessaire pour un grand nombre de stations de travail. Cela peut être dangereux car vous devez autoriser le serveur de fichiers à se connecter aux clients (le serveur X d'un point de vue X. X intervertit la notion de

client et de serveur). Si vous suivez les (très mauvaises) suggestions de nombreuses documentations, vous tapez `xhost +` sur la machine. Cela autorise tout client X à se connecter au système. Pour une sécurité légèrement meilleure, vous pouvez utiliser la commande `xhost +hostname` à la place, ce qui permet de n'autoriser les accès que depuis certains hôtes.

Une solution encore meilleure serait d'utiliser un tunnel SSH pour X et de chiffrer toute la session. C'est fait automatiquement lors de l'utilisation de SSH pour se connecter sur une autre machine. Pour que cela fonctionne, vous devez configurer à la fois le client SSH et le serveur SSH. Sur le client SSH, `ForwardX11` doit être positionné à `yes` dans `/etc/ssh/ssh_config`. Sur le serveur SSH, `X11Forwarding` doit être positionné à `yes` dans `/etc/ssh/sshd_config` et le paquet xbase-clients doit être installé car le serveur SSH utilise `/usr/X11R6/bin/xauth` (`/usr/bin/xauth` sur Debian unstable) pour mettre en place le pseudoaffichage X. À l'heure de SSH, vous devriez abandonner complètement le contrôle d'accès basé sur xhost.

Pour une sécurité accrue, si vous n'avez pas besoin d'accéder à X depuis d'autres machines, désactivez l'écoute sur le port TCP 6000 en tapant simplement :

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

C'est le comportement par défaut dans XFree 4.1.0 (le serveur X fourni dans Debian 3.0 et 3.1). Si vous utilisez XFree 3.3.6 (vous avez donc Debian 2.2 installée), vous pouvez éditer `/etc/X11/xinit/xserverrc` afin d'avoir quelque chose ressemblant à ceci :

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

Si vous utilisez XDM, mettez `/etc/X11/xdm/Xservers` à : `:0 local /usr/bin/X11/X vt7 -dpi 100 -nolisten tcp`. Si vous utilisez GDM, assurez-vous que l'option `DisallowTCP=true` est positionnée dans `/etc/gdm/gdm.conf` (qui est par défaut dans Debian). Cela va basiquement ajouter `-nolisten tcp` à chaque ligne de commande X [1].

Vous pouvez également positionner l'expiration de délai système par défaut pour les blocages **xscreensaver**. Même si l'utilisateur peut annuler cela, vous devriez éditer le fichier de configuration `/etc/X11/app-defaults/XScreenSaver` et changer la ligne de blocage :

```
*lock:                  False
```

(qui est par défaut dans Debian) à :

```
*lock:                  True
```

FIXME : Ajouter des informations sur comment désactiver les économiseurs d'écran qui affichent l'écran de l'utilisateur (qui peuvent avoir des informations sensibles).

Plus de renseignements sur la sécurité X Window dans http://www.linuxdoc.org/HOWTO/XWindow-User-HOWTO.html (`/usr/share/doc/HOWTO/en-txt/XWindow-User-HOWTO.txt.gz`).

FIXME : Ajouter des informations d'une discussion de debian-security pour avoir les modifications des fichiers de configuration de XFree 3.3.6 pour faire cela.

[1] GDM *n'ajoutera pas* `-nolisten tcp` s'il trouve `-query` ou `-indirect` sur la ligne de commande car cela ne pourrait pas fonctionner.

Vérifiez le gestionnaire d'affichage

Si vous ne voulez un gestionnaire d'affichage installé que pour une utilisation locale (avec une jolie connexion graphique, tout de même), assurez-vous que le XDMCP (X Display Manager Control Protocol) est désactivé. Dans XDM, vous pouvez faire cela avec cette ligne dans `/etc/X11/xdm/xdm-config` :

```
DisplayManager.requestPort:     0
```

Pour GDM, il devrait y avoir dans le fichier gdm.conf :

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

Normalement, tous les gestionnaires d'affichages sont configurés par défaut pour ne pas démarrer les services XDMCP dans Debian.

Sécurisation de l'accès à l'impression (le problème lpd et lprng)

Imaginez, vous arrivez au travail et l'imprimante crache une quantité infinie de papier car quelqu'un est en train de provoquer un déni de service sur le démon d'impression. Méchant, n'est ce pas ?

Dans toute architecture d'impression UNIX, il y a un moyen de fournir les données du client vers le serveur d'impression de l'hôte. Dans les traditionnels **lpr** et **lp**, la commande du client copie ou crée un lien symbolique pour les données dans le répertoire de spool (c'est pour cela que ces programmes sont habituellement SUID ou SGID).

Pour éviter tout problème, vous devriez garder vos serveurs d'impression particulièrement sûrs. Cela veut dire qu'il est nécessaire de configurer le service d'impression pour qu'il autorise seulement les connexions d'un ensemble de serveurs de confiance. Pour ce faire, ajoutez les serveurs auxquels vous voulez autoriser l'impression à `/etc/hosts.lpd`.

Cependant, même si vous faites cela, le démon **lpr** accepte les connexions entrantes sur le port 515 de n'importe quelle interface. Vous devriez réfléchir au filtrage par un pare-feu des connexions provenant de réseaux ou hôtes qui ne sont pas autorisés à imprimer (le démon **lpr** ne peut être limité que pour écouter sur une adresse IP donnée).

lprng doit être préféré à **lpr** car il peut être configuré pour faire du contrôle d'accès basé sur l'adresse IP. Vous pouvez indiquer l'interface sur laquelle se lier (cependant d'une manière un peu bizarre)

Si vous utilisez une imprimante sur le système, mais seulement localement, vous ne voulez pas partager ce service sur le réseau. Vous pouvez considérer l'utilisation d'autres systèmes d'impression, comme celui fourni par cups ou http://pdq.sourceforge.net/ qui est basé sur les permissions utilisateurs du périphérique `/dev/lp0`.

Dans cups, les données d'impression sont transférées vers le serveur par le protocole HTTP. Cela veut dire que le programme client n'a pas besoin de privilèges spéciaux, mais cela nécessite que le serveur écoute sur un port quelque part.

Cependant, si vous voulez utiliser **cups**, mais seulement localement, vous pouvez le configurer pour se lier à l'interface de bouclage (loopback) en modifiant `/etc/cups/cupsd.conf` :

```
Listen 127.0.0.1:631
```

Il y a plusieurs autres options de sécurité comme autoriser ou interdire des réseaux et hôtes dans le fichier de configuration. Cependant, si vous n'en avez pas besoin, il peut être préférable de simplement limiter le port d'écoute. **cups** fournit également la documentation par le port HTTP, si vous ne voulez pas dévoiler des informations potentiellement utiles aux attaquants extérieurs (et que le port est ouvert), ajoutez également :

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

Ce fichier de configuration peut être modifié pour ajouter plus de fonctionnalités y compris des certificats SSL/TLS et du chiffrement. Les manuels sont disponibles sur http://localhost:631/ ou à cups.org.

FIXME : Ajouter plus de contenu (l'article sur http://www.rootprompt.org fournit certains points de vues très intéressants).

FIXME : Vérifier la disponibilité de PDG dans Debian, et s'il l'est, le suggérer comme le système d'impression préféré.

FIXME : Vérifier si Farmer/Wietse a une alternative pour le démon d'imprimante et si il est disponible dans Debian.

Sécurisation du service de courrier

Si le serveur n'est pas un système d'envoi de courrier, vous n'avez pas réellement besoin d'un démon de courrier écoutant les connexions entrantes, mais vous pourriez vouloir que le courrier local soit distribué pour, par exemple, recevoir le courrier du superutilisateur en provenance d'un des systèmes d'alerte en place.

Si vous avez **exim**, vous n'avez pas besoin que le démon tourne pour le faire car la tâche standard **cron** vide la file des messages. Consultez la section intitulée « Désactivation de services démon » pour le façon de faire cela.

Configurer un Nullmailer

Vous pouvez vouloir avoir un démon local de courrier pour qu'il puisse relayer les courriers envoyés localement à un autre système. C'est courant quand vous devez administrer un certain nombre de systèmes et que vous ne voulez pas vous connecter à chacun d'entre eux pour lire le courrier envoyé localement. Comme toute la journalisation de chaque système individuel peut être centralisée en utilisant un serveur de journalisation système centralisé, les courriers peuvent être envoyés à un serveur de courriers central.

Un tel système *relais seulement* devrait être configuré correctement pour cela. Le démon pourrait également être configuré pour n'écouter que sur l'adresse de bouclage.

Les étapes de configuration suivantes ne doivent être suivies que si vous configurez le paquet exim dans la version 3.0 de Debian. Si vous utilisez une version ultérieure (comme la version 3.1 qui utilise exim4), le système d'installation a été amélioré afin, si le MTA est configuré pour ne délivrer que des messages locaux, de n'autoriser des connexions que depuis l'hôte local et interdire toute connexion distante.

Sur un système Debian 3.0 utilisant exim, vous devrez retirer le démon SMTP d'inetd :

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

et configurer le démon de courrier pour écouter seulement sur l'interface de bouclage. Dans **exim** (le MTA par défaut) vous pouvez faire ça en éditant le fichier `/etc/exim.conf` et en ajoutant la ligne suivante :

```
local_interfaces = "127.0.0.1"
```

Redémarrez les deux démons (inetd et exim) et Exim n'écoutera que sur la socket 127.0.0.1:25. Faites attention, et avant tout désactivez inetd, sinon Exim ne démarrera pas étant donné que le démon inetd est déjà en attente de connexions entrantes.

Pour **postfix** éditez `/etc/postfix/main.conf` :

```
inet_interfaces = localhost
```

Si vous voulez seulement le courrier local, cette approche est meilleure que l'encapsulation TCP du démon de courrier ou l'ajout de règles de pare-feu pour limiter les personnes qui y accèdent. Cependant, si vous n'avez pas besoin d'écouter sur d'autres interfaces, vous pourriez envisager de le lancer à partir d'inetd et ajouter une encapsulation TCP pour que les connexions entrantes soient vérifiées par rapport à `/etc/hosts.allow` et `/etc/hosts.deny`. De plus, vous serez au courant quand un accès non autorisé est tenté sur le démon de courrier, si vous mettez en place correctement la journalisation pour n'importe laquelle des méthodes décrites plus haut.

En tout cas, pour rejeter les tentatives de relais de courrier au niveau SMTP, vous pouvez modifier `/etc/exim/exim.conf` pour inclure :

```
receiver_verify = true
```

Même si le serveur de courrier ne relaiera pas le message, ce genre de configuration est nécessaire au testeur de relais à http://www.abuse.net/relay.html pour déterminer que le serveur ne peut *pas* faire de relais.

Si vous voulez une configuration relais seulement, cependant, vous pouvez vouloir changer le démon de courrier pour des programmes qui ne peuvent être configurés *que* pour faire suivre le courrier à un serveur de courrier distant. Debian fournit actuellement les paquets ssmtp et nullmailer dans ce but. En tout cas, vous pouvez évaluer pour vous-même l'un de ces deux agents de transport de courrier [2]La liste n'inclura pas **qmail**, qui est distribué seulement comme code source dans le paquet qmail-src fournis par Debian et voir lequel correspond le mieux aux buts du système.

Fournir un accès sécurisé aux boîtes à lettres

Si vous désirez donner un accès à distance aux boîtes à lettres, il y a un certain nombre de démons POP3 et IMAP disponibles[3] Cependant, si vous fournissez un accès IMAP, notez qu'il s'agit d'un protocole générique d'accès aux fichiers, il peut devenir l'équivalent d'un accès à l'interpréteur de commandes car les utilisateurs peuvent être capables de récupérer n'importe quel fichier par celle-ci.

Essayez, par exemple, de configurer comme chemin de votre boîte de réception `{server.com}/etc/passwd`, si cela réussit, votre démon IMAP n'est pas configuré correctement pour empêcher ce genre d'accès.

[2] Pour récupérer la liste des démons de courrier disponibles dans Debian, essayez :

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

[3] Une liste des serveurs et démons prenant ces protocoles en charge dans Debian peut être récupérée avec :

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

Parmi les serveurs IMAP dans Debian, le serveur **cyrus** (dans le paquet cyrus-imapd) contourne cela en ayant tous les accès sur une base de données dans une partie restreinte du système de fichiers. Également, **uw-imapd** (installez soit uw-imapd ou mieux, si votre client IMAP le gère, uw-imapd-ssl) peut être configuré pour « chrooter » les répertoires de courrier des utilisateurs, mais cela n'est pas activé par défaut. La documentation fournie donne plus d'informations sur la façon de le configurer.

Vous pouvez également vouloir faire fonctionner un serveur IMAP qui n'ait pas besoin que des utilisateurs valables soient créés sur le système local (ce qui donnerait également un accès à l'interpréteur de commande), les paquets courier-imap (pour IMAP), courier-pop teapop (pour POP3) et cyrus-imapd (pour POP3 et IMAP) fournissent des serveurs avec des méthodes d'authentification en plus des comptes utilisateur locaux. **cyrus** peut utiliser toute méthode d'authentification qui peut être configurée par PAM tandis que **teapop** peut utiliser des bases de données (comme postgresql et mysql) pour l'authentification des utilisateurs.

FIXME : Vérifier : uw-imapd peut être configuré avec l'authentification utilisateur grâce à PAM également.

Réception du courrier de manière sûre

La lecture et réception du courrier sont des protocoles en texte clair parmi les plus courants. Si vous utilisez POP3 ou IMAP pour récupérer le courrier, vous envoyez votre mot de passe en clair à travers le réseau, et donc presque tout le monde peut lire votre courrier à partir de maintenant. À la place, utilisez SSL (Secure Sockets Layer) pour recevoir votre courrier. L'autre alternative est SSH, si vous avez un compte avec interpréteur de commandes sur la machine qui sert de serveur POP ou IMAP. Voici un `fetchmailrc` simple décrivant cela :

```
poll mon-serveur-imap.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
     mon-serveur-imap.org sleep 15 </dev/null > /dev/null'
```

Le preconnect est la ligne importante. Il lance une session SSH et crée le tunnel nécessaire, qui relaie automatiquement les connexions au port local 1236 vers le port IMAP du serveur de mail, mais chiffrées. Une autre possibilité serait d'utiliser **fetchmail** avec la fonctionnalité SSL.

Si vous désirez fournir des services de courrier comme POP et IMAP chiffrés, `apt-get install stunnel` et démarrez vos démons ainsi :

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

Cette commande encapsule le démon fourni (-l) au port (-d) et utilise le certificat SSL indiqué (-p).

Sécurisation de BIND

Il y a différents problèmes qui peuvent être traités pour sécuriser le démon de serveur de domaine; problèmes similaires à ceux étudiés quand on sécurise n'importe quel service donné :

- configurer le démon lui-même pour qu'il ne puisse pas être mal utilisé de l'extérieur (consultez la section intitulée « Configuration de BIND pour éviter de mauvaises utilisations »). Cela inclut limiter les requêtes possibles pour les clients : transferts de zones et requêtes récursives ;

- limiter l'accès du démon au serveur lui-même, ainsi s'il est utilisé pour s'introduire, les dommages au système sont limités. Cela inclut d'exécuter le démon en tant qu'utilisateur non privilégié (consultez la section intitulée « Changer l'utilisateur de BIND ») et le chrooter (consultez la section intitulée « Chrooter le serveur de domaine »).

Configuration de BIND pour éviter de mauvaises utilisations

Vous devriez restreindre certains renseignements donnés par le serveur DNS aux clients extérieurs pour l'empêcher d'être utilisé pour obtenir des informations de valeur sur votre organisation que vous ne voudriez pas divulguer. Cela inclut l'ajout des options suivantes : *allow-transfer*, *allow-query*, *allow-recursive* et *version*. Vous pouvez soit limiter cela dans la section globale (pour que cela s'applique à toutes les zones servies) ou individuellement par zone. Cette information est documentée dans le paquet bind-doc, consultez `/usr/share/doc/bind/html/index.html` en plus à ce sujet une fois que le paquet est installé.

Imaginez que votre serveur (un serveur avec plusieurs adresses de base) est connecté à Internet et à votre réseau interne (votre adresse IP interne est 192.168.1.2), vous ne voulez fournir aucun service à Internet et vous voulez juste autoriser les consultations DNS à partir de vos hôtes internes. Vous pourriez le restreindre en incluant dans `/etc/bind/named.conf`:

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

L'option *listen-on* lie uniquement le DNS à l'interface ayant une adresse interne, mais, même si cette interface est la même que l'interface qui permet la connexion à Internet (par l'utilisation de NAT, par exemple), les requêtes ne seront acceptées que si celles-ci proviennent d'hôtes internes. Si le système est constitué de plusieurs interfaces et que le *listen-on* n'est pas présent, seuls les utilisateurs internes pourront émettre des requêtes, mais, puisque le port restera accessible à des attaquants externes, ils pourront essayer de faire tomber (ou exploiter une attaque de débordement de tampon sur) le serveur DNS. Vous pouvez même le mettre uniquement en écoute sur l'adresse 127.0.0.1 si vous ne désirez offrir le service à personne d'autre que vous même.

L'enregistrement version.bind dans la classe chaos contient la version du processus bind actuellement en cours d'exécution. Cette information est souvent utilisée par des scanners automatisés et des individus malveillants qui souhaitent déterminer si un **bind** est vulnérable à une attaque spécifique. En fournissant des informations fausses ou pas d'informations du tout, on limite la probabilité qu'un serveur soit attaqué sur la base de la version qu'il publie. Pour fournir votre propre version, utilisez la directive *version* de la manière suivante :

```
options {
        ... diverses options ici ...
        version "Not available.";
 };
```

Changer l'enregistrement version.bind ne fournit pas actuellement de protection contre les attaques, mais cela devrait être considéré comme une protection utile.

Un fichier de configuration `named.conf` d'exemple pourrait être me suivant :

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

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

options {
        directory "/var/cache/bind";
        allow-query { internal; };
        allow-recursion { internal; };
        allow-transfer { none; };
};
// À partir d'ici jusqu'à la zone mysite.bogus
// est dans l'ensemble non modifié des valeurs par défaut 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 ajoutées moi-même
zone "mysite.bogus" {
        type master;
        file "/etc/bind/named.mysite";
        allow-query { any; };
        allow-transfer { friendly; };
};
```

Veuillez vérifier (de nouveau) le système de suivi des bogues (BTS) à propos de BIND, en particulier le http://bugs.debian.org/94760. Vous pouvez contribuer si vous le désirez au rapport de bogue si vous pensez pouvoir ajouter des informations utiles.

Changer l'utilisateur de BIND

Concernant la limitation des privilèges de BIND vous devez être conscient que si un utilisateur différent du superutilisateur exécute BIND, alors BIND ne peut pas détecter de nouvelles interfaces automatiquement, par exemple, quand vous insérez une carte PCMCIA dans un portable. Consultez le fichier `README.Debian` du répertoire de documentation de named (`/usr/share/doc/bind/README.Debian`) pour plus d'informations sur ce problème. De nombreux problèmes de sécurité concernant BIND ont été récemment découverts, donc le changement d'utilisateur est utile si possible, cependant si vous désirez le faire de façon automatique, vous pouvez essayer le script fourni dans la section intitulée « Exemple de script pour changer l'installation par défaut de BIND ».

Remarquez, de toute façon, que cela ne concerne que la version 8 de BIND. Dans les paquets Debian de la version 9 (depuis la version 9.2.1-5, disponible avec Sarge), l'utilisateur *bind* est créé et utilisé en configurant la variable OPTIONS de `/etc/default/bind9`. Si vous utilisez BIND version 9 et que le démon de serveur de noms ne fonctionne pas avec l'utilisateur *bind*, vérifiez les configurations de ce fichier.

Pour démarrer BIND sous un autre utilisateur, tout d'abord créez un utilisateur et un groupe séparé (ce n'est *pas* une bonne idée d'utiliser nobody ou nogroup pour chaque service ne devant pas tourner en tant que superutilisateur). Dans cet exemple, l'utilisateur et le groupe `named` seront utilisés. Vous pouvez faire cela en tapant :

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

Notez que l'utilisateur `named` sera très restreint. Si vous désirez, pout toute raison, avoir une configuration moins restrictive, utilisez :

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

Maintenant vous pouvez soit éditer, à l'aide de votre éditeur favori, `/etc/init.d/bind` et modifiez les lignes commençant par

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

en[4]

[4] Notez que selon la version de BIND, l'option `-g` risque de ne pas être disponible, en particulier si vous utilisez bind9 avec *Sarge* (version 9.2.4).

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

soit modifier (créez-le s'il n'existe pas) le fichier de configuration par défaut (`/etc/default/bind` pour BIND en version 8) et introduisez ceci :

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

Modifiez les permissions des fichiers utilisés par BIND, y compris `/etc/bind/rndc.key` :

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

et l'endroit où BIND crée son fichier pid en utilisant, par exemple `/var/run/named` au lieu de `/var/run` :

```
$ mkdir /var/run/named
$ chown named.named /var/run/named
$ vi /etc/named.conf
[ ... mettez le fichier de configuration à jour en utilisant ce nouvel
emplacement ...]
options { ...
        pid-file "/var/run/named/named.pid";
};
[ ... ]
```

Pour éviter également d'exécuter quoi que ce soit en tant que superutilisateur, modifiez la ligne `reload` du script init.d en substituant :

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

par :

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

Remarque : selon la version de Debian, vous pouvez devoir changer la ligne `restart` également. Cela a été corrigé dans la version `1:8.3.1-2` de BIND pour Debian.

Il ne reste plus qu'à redémarrer BIND à l'aide de `/etc/init.d/bind restart`, puis rechercher dans le journal système les deux entrées suivantes :

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

Voilà ! Maintenant named ne s'exécute *plus* en tant que superutilisateur. Si vous voulez lire plus d'informations sur pourquoi BIND ne fonctionne pas en tant qu'utilisateur non superutilisateur sur les systèmes Debian, veuillez vérifier le système de suivi des bogues concernant BIND, en particulier les bogues http://bugs.debian.org/50013, http://bugs.debian.org/132582, http://bugs.debian.org/53550, http://

bugs.debian.org/52745 et http://bugs.debian.org/128129. Vous pouvez contribuer à ces rapports de bogue si vous le désirez si vous pensez pouvoir ajouter des informations utiles.

Chrooter le serveur de domaine

Pour atteindre une sécurité de BIND maximale, construisez maintenant une prison chroot (consultez la section intitulée « Paranoïa généralisée du suid et du chroot ») autour du démon. Il y a un moyen facile de faire cela : l'option `-t` (consultez la page de manuel named(8) ou la page 100 de la http://www.nominum.com/content/documents/bind9arm.pdf). Cela fera que BIND se chrootera lui-même dans le répertoire donné sans que vous ayez besoin de configurer une prison chroot et de vous inquiéter au sujet des bibliothèques dynamiques. Les seuls fichiers qui doivent être dans cette prison chroot :

```
dev/null
etc/bind/       - doit contenir named.conf et toutes les zones du serveur
sbin/named-xfer - si vous faites du transfert de nom
var/run/named/  - devrait contenir le PID et le cache du serveur de nom
                  (s'il existe), ce répertoire doit être accessible en
                  écriture à l'utilisateur named
var/log/named   - si vous configurez le journal vers un fichier, doit
                  être accessible en écriture à l'utilisateur named
dev/log         - syslogd devrait écouter ici si named est configuré
                  pour journaliser en l'utilisant
```

Pour que le démon BIND fonctionne correctement il a besoin de permissions dans les fichiers named. C'est une tâche facile car les fichiers de configuration sont toujours dans `/etc/named`. Prenez en compte qu'il n'a besoin que d'un accès en lecture seule aux fichiers de zone, sauf s'il s'agit un serveur de nom secondaire ou serveur cache. Si c'est le cas vous devrez permettre la lecture et l'écriture aux zones nécessaires (pour que les transferts de zone à partir du serveur primaire fonctionnent).

De plus, vous pouvez trouver plus d'informations concernant le chrootage de BIND dans le http://www.linuxdoc.org/HOWTO/Chroot-BIND-HOWTO.html (au sujet de BIND 9) et http://www.linuxdoc.org/HOWTO/Chroot-BIND8-HOWTO.html (au sujet de BIND 8). Ces mêmes documents devraient être disponibles par l'installation de doc-linux-text (version texte) ou doc-linux-html (version HTML). Un autre document utile est http://web.archive.org/web/20011024064030/http://www.psionic.com/papers/dns/dns-linux.

Si vous configurez une véritable prison chroot (c'est-à-dire pas seulement l'option `-t`) pour BIND dans Debian, assurez-vous qu'elle contient les fichiers suivants[5] :

```
dev/log - syslogd devrait écouter ici
dev/null
etc/bind/named.conf
etc/localtime
etc/group - avec une seule ligne: "named:x:GID:"
etc/ld.so.cache - généré avec ldconfig
lib/ld-2.3.6.so
lib/libc-2.3.6.so
lib/ld-linux.so.2 - lié symboliquement à ld-2.3.6.so
lib/libc.so.6 - lié symboliquement à libc-2.3.6.so
sbin/ldconfig - pourra être effacé après la configuration du chroot
```

[5] Cette configuration n'a pas encore été essayée pour les nouvelles versions de BIND.

```
sbin/named-xfer - si vous faites des transferts de nom
var/run/
```

Modifiez aussi l'écoute de **syslogd** sur `$CHROOT/dev/log` pour que le serveur de nom puisse écrire des entrées de journalisation système dans le journal du système local.

Pour éviter des problèmes avec les bibliothèques dynamiques, vous pouvez compiler BIND statiquement. Vous pouvez utiliser **apt-get** pour cela avec l'option `source`. Il peut même récupérer les paquets dont vous avez besoin pour le compiler correctement. Il vous faudrait faire quelque chose comme :

```
$ apt-get source bind
# apt-get build-dep bind
$ cd bind-8.2.5-2
  (modifier src/port/linux/Makefile pour que CFLAGS contienne
   l'option « -static »)
$ dpkg-buildpackage -rfakeroot -uc -us
$ cd ..
# dpkg -i bind-8.2.5-2*deb
```

Après l'installation, vous devrez déplacer des fichiers dans la prison chroot[6] vous pouvez conserver les scripts `init.d` dans `/etc/init.d` pour que le système lance automatiquement le serveur de domaine, mais éditez les pour ajouter `--chroot /location_of_chroot` dans les appels à **start-stop-daemon** dans ces scripts ou utilisez l'option *-t* de BIND en la configurant dans l'argument OPTIONS du fichier de configuration `/etc/default/bind` (pour la version 8) ou `/etc/default/bind9` (pour la version 9).

Pour plus d'informations sur la mise en place de chroots, consultez la section intitulée « Paranoïa généralisée du suid et du chroot ».

FIXME : Inclure les informations provenant de http://people.debian.org/~pzn/howto/chroot-bind.sh.txt, http://www.cryptio.net/~ferlatte/config/ (spécifique Debian), http://web.archive.org/web/20021216104548/http://www.psionic.com/papers/whitep01.html, http://csrc.nist.gov/fasp/FASPDocs/NISTSecuringDNS.htm.

Sécurisation d'Apache

FIXME : Ajout de contenu : modules fournis par l'installation normale d'Apache (sous /usr/lib/apache/X.X/mod_*) et modules qui peuvent être installés séparément dans les paquets libapache-mod-XXX.

Vous pouvez limiter l'accès au serveur Apache si vous voulez uniquement l'utiliser en interne (dans un but d'essai, pour accéder à l'archive doc-central, etc.) et si vous ne voulez pas que des intrus y accèdent. Pour réaliser cela, utilisez les directives `Listen` ou `BindAddress` dans `/etc/apache/http.conf`.

En utilisant Listen :

```
Listen 127.0.0.1:80
```

En utilisant BindAddress :

```
BindAddress 127.0.0.1
```

[6] Sauf si vous utilisez l'option `instdir` lors de l'appel à **dpkg** mais alors la prison chroot peut être un petit peu plus complexe.

Ensuite, redémarrez apache avec `/etc/init.d/apache restart` et vous observerez qu'il écoute uniquement l'interface loopback.

Dans tous les cas, si vous n'utilisez pas toutes les fonctionnalités fournies par Apache, vous pouvez jeter un œil aux autres serveurs web fournis dans Debian comme dhttpd.

La http://httpd.apache.org/docs/misc/security_tips.html fournit des informations concernant les mesures de sécurité à prendre pour les serveurs web Apache (ces mêmes informations sont fournies dans Debian par le paquet apache-doc).

Plus d'informations sur des restrictions supplémentaires d'Apache en mettant en place une prison chrooté sont disponibles en la section intitulée « Environnement de chroot pour Apache ».

Désactiver la publication de contenu sur le web par les utilisateurs

L'installation par défaut d'Apache dans Debian permet aux utilisateurs de publier du contenu dans leur répertoire `$HOME/public_html`. Ce contenu peut être récupéré à distance en utilisant une URL comme : http://serveur_apache/~utilisateur.

Pour empêcher cela, veuillez modifier le fichier de configuration `/etc/apache/http.conf` en commentant (pour Apache 1.3) le module suivant :

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

Avec Apache 2.0, il faut supprimer le fichier `/etc/apache2/mods-enabled/userdir.load` ou restreindre la configuration par défaut en modifiant `/etc/apache2/mods-enabled/userdir.conf`.

Cependant, si le module a été lié statiquement (vous pouvez obtenir la liste des modules compilés en exécutant `apache -l`), vous devez ajouter la ligne suivante au fichier de configuration d'Apache :

```
Userdir disabled
```

Un attaquant peut encore faire de l'énumération d'utilisateur, car la réponse du serveur web sera un *403 Permission Denied* et non un *404 Not available*. Vous pouvez éviter cela en utilisant le module Rewrite.

Permissions des fichiers de journalisation

Les fichiers de journalisation d'Apache, depuis la version 1.3.22-1, ont pour propriétaire l'utilisateur « root » et pour groupe « adm » avec les permissions 640. Ces permissions sont changées après la rotation. Un intrus qui peut accéder au système par le serveur web ne pourra pas (sans augmentation de droits) enlever d'anciennes entrées de fichiers de log.

Fichiers web publiés

Les fichiers d'Apache sont situés sous `/var/www`. Juste après l'installation, le fichier par défaut fournit quelques informations sur le système (principalement qu'il s'agit d'un système Debian exécutant Apache). Les pages web par défaut appartiennent à l'utilisateur root et au groupe root par défaut alors que le processus Apache s'exécute avec l'utilisateur www-data et le groupe www-data. Cela devrait rendre plus difficile aux attaquants qui compromettent le système par le site web de le défigurer. Vous devriez, bien sûr, remplacer les pages web par défaut (qui peuvent fournir des informations que vous ne voulez pas donner aux visiteurs) avec les vôtres.

Sécurisation de finger

Si vous désirez utiliser le service finger, demandez-vous si vous en avez réellement besoin. Si oui, vous découvrirez que Debian fournit de nombreux démons finger (sortie d'un **apt-cache search fingerd**):

- cfingerd - démon finger configurable
- efingerd - autre démon finger pour UNIX capable de syntoniser précisément la sortie
- ffingerd - démon finger sécurisé
- fingerd - serveur distant pour informations d'utilisateurs
- xfingerd - démon finger de type BSD avec la prise en charge de qmail

ffingerd est le démon finger recommandé si vous comptez l'utiliser pour un service public. Dans tous les cas, vous devriez, lors de la mise en place par inetd, xinetd ou tcpserver, limiter le nombre de processus qui seront lancés en même temps, limiter les accès au démon finger depuis un nombre d'hôtes donné (en utilisant l'encapsulation TCP) et l'avoir en écoute uniquement sur une interface bien définie.

Paranoïa généralisée du suid et du chroot

chroot est l'une des plus puissantes possibilités pour restreindre un démon, un utilisateur ou un autre service. Imaginez simplement une prison autour de votre cible, de laquelle votre cible ne peut s'échapper (normalement, mais il y a encore beaucoup de conditions qui peuvent permettre de s'échapper d'une telle prison). Si vous ne faites pas confiance à l'utilisateur ou au service, vous pouvez créer un environnement racine modifié pour lui. Cela peut utiliser pas mal d'espace disque car vous devez copier tous les exécutables nécessaires, ainsi que des bibliothèques, dans la prison. Mais alors, même si l'utilisateur fait quelque chose de malveillant, l'étendue des dommages est limitée à la prison.

Un grand nombre de services fonctionnant en démons pourraient bénéficier de ce type d'arrangement. Les démons que vous installez dans votre distribution Debian ne seront cependant pas fournis chrootés[7] par défaut.

Exemples : serveurs de noms de domaine (comme **bind**), serveurs web (comme **apache**), serveurs de courrier (comme **sendmail**) et serveurs FTP (comme **wu-ftpd**). La complexité de BIND est probablement la raison pour laquelle il a été exposé à de nombreuses attaques ces dernières années (consultez la section intitulée « Sécurisation de BIND »).

Cependant, Debian fournit des logiciels qui peuvent vous aider à mettre en place des environnements **chroot**. Consultez la section intitulée « Créer des environnements chrooté automatiquement ».

De toute façon, si vous exécutez un quelconque service sur votre système, vous devriez considérer de le faire fonctionner de la façon la plus sécurisée possible. Cela comprend : révoquer les droits du superutilisateur, le faire fonctionner dans un environnement restreint (comme une prison chroot) ou le remplacer par un équivalent plus sécurisé.

Cependant, soyez prévenu qu'une prison **chroot** peut être cassée si l'utilisateur fonctionnant dedans est le superutilisateur. Vous devez donc faire fonctionner le service avec un utilisateur sans droits élevés. En limitant son environnement, vous limitez les fichiers lisibles et exécutables par tout le monde auxquels le service peut accéder, vous limitez donc aussi les possibilités d'une augmentation de droits en utilisant des

[7] Elle essaie de les faire fonctionner avec le *minimum de droits*, y compris exécuter les démons avec leur propre utilisateur au lieu de les exécuter en tant que superutilisateur.

failles de sécurité sur le système local. Même dans une situation où vous ne pouvez pas être complètement certain qu'il n'y a pas de moyen pour un attaquant intelligent de sortir de la prison d'une manière ou d'une autre. Utiliser seulement des programmes serveur ayant une réputation de sécurité est une bonne mesure de sécurité additionnelle. Même des trous minuscules comme des descripteurs de fichier peuvent être utilisés par un attaquant doué pour s'introduire dans le système. Après tout, **chroot** n'a pas été conçu pour être un outil de sécurité, mais un outil de test.

Créer des environnements chrooté automatiquement

Plusieurs programmes permettent de chrooter automatiquement des serveurs et services. Debian fournit actuellement (accepté en mai 2002) **chrootuid** de Wietse Venema dans le paquet chrootuid, ainsi que compartment et makejail. Ces programmes peuvent être utilisés pour mettre en place un environnement restreint pour exécuter tout programme (**chrootuid** vous permet même de l'exécuter avec un utilisateur restreint).

Certains de ces outils peuvent être utilisés pour mettre en place l'environnement chrooté facilement. Le programme **makejail**, par exemple, peut créer et mettre à jour une prison chroot avec de petits fichiers de configuration (il fournit des fichiers de configuration exemple pour **bind**, **apache**, **postgresql** et **mysql**). Il tente de deviner et d'installer dans la prison tous les fichiers nécessaires au démon en utilisant **strace**, **stat** et les dépendances du paquet Debian. De plus amples renseignements sont disponibles à http://www.floc.net/makejail/. **Jailer** est un outil semblable disponible à http://www.balabit.hu/downloads/jailer/ et en paquet Debian.

Paranoïa généralisée du mot de passe en texte clair

Vous devriez essayer d'éviter tout service réseau qui envoie et reçoit des mots de passe en texte clair par le net comme FTP/TELNET/NIS/RPC. L'auteur recommande l'utilisation de SSH à la place de TELNET et FTP pour tout le monde.

Gardez à l'esprit que la migration de TELNET vers SSH, en conservant l'utilisation d'autres protocoles à texte non chiffrés n'augmente votre sécurité en AUCUNE manière ! Le mieux serait de retirer FTP, TELNET, POP, IMAP, HTTP et de les remplacer par leurs services chiffrés respectifs. Vous devriez considérer la migration de ces services vers leurs versions SSL, ftp-ssl, telnet-ssl, pop-ssl, HTTPS, etc.

La plupart des astuces ci-dessus s'appliquent à tout système UNIX (vous les trouverez dans des documents de durcissement liés à Linux et autres UNIX).

Désactivation du NIS

Si possible, évitez d'utiliser NIS, le service d'informations réseau (« Network Information Service »), car il autorise le partage de mot de passe. Cela peut être fortement dangereux si votre installation est cassée.

Si vous avez besoin de partager les mots de passe entre machines, pensez à d'autres alternatives. Par exemple, mettre en place un serveur LDAP et configurer PAM sur votre système afin de contacter le serveur LDAP pour l'authentification des utilisateurs. Une installation détaillée est disponible dans le http://www.linuxdoc.org/HOWTO/LDAP-HOWTO.html (`/usr/share/doc/HOWTO/en-txt/LDAP-HOWTO.txt.gz`).

Des informations supplémentaires sur la sécurité de NIS sont disponibles à l'adresse http://www.linuxdoc.org/HOWTO/NIS-HOWTO.html (`/usr/share/doc/HOWTO/fr-txt/NIS-HOWTO.txt.gz`).

FIXME (jfs) : Ajouter des renseignements sur la façon de configurer cela dans Debian.

Sécurisation des services RPC

Vous devriez désactiver RPC si vous n'en avez pas besoin.

Les appels de procédure à distance (« Remote Procedure Call » ou RPC) sont un protocole que les programmes peuvent utiliser pour demander des services de la part d'autres programmes liées sur différents ordinateurs. Le service **portmap** contrôle les services RPC en convertissant les numéros de programme RPC en numéros de port du protocole DARPA ; il doit fonctionner pour pouvoir faire des appels RPC.

Les services basés sur RPC ont eu un mauvaise historique de trous de sécurité, bien que le portmapper lui-même n'en a pas (mais il fournit des informations à un attaquant distant). Notez que certaines des attaques DDoS (déni de service distribué) exploitent RPC pour entrer dans le système et agir en tant qu'agent ou gestionnaire.

Vous n'avez besoin de RPC que si vous utilisez un service basé sur RPC. Les services basés sur RPC les plus communs sont NFS (Network File System) et NIS (Network Information System). Consultez la section précédente pour plus d'informations à propos de NIS. Le File Alteration Monitor (FAM) fourni par le paquet fam est également un service RPC et dépend donc de portmap.

Les services NFS sont assez importants dans certains réseaux. Si c'est le cas pour vous, vous aurez alors besoin de trouver un équilibre entre la sécurité et l'utilisabilité du réseau (plus de renseignements à propos de la sécurité NFS sont disponibles dans le http://www.tldp.org/HOWTO/NFS-HOWTO.html ou `/usr/share/doc/HOWTO/fr-txt/NFS-HOWTO.txt.gz`).

Désactivation des services RPC

La désactivation de portmap est assez simple. Il y a différentes méthodes. La plus simple sur un système Debian 3.0 et versions supérieures est de désinstaller le paquet portmap. Si vous exécutez une version plus ancienne, vous devrez désactiver le service comme expliqué dans la section intitulée « Désactivation de services démon », cela est dû au fait que le programme fait partie du paquet netbase (qui ne peut être désinstallé sans endommager le système).

Notez que certains environnements de bureau (notamment, GNOME) utilisent des services RPC et ont besoin du portmapper pour certaines fonctionnalités de gestion de fichiers. Si c'est votre cas, vous pouvez limiter l'accès aux services RPC comme décrit ci-dessous.

Limiter l'accès aux services RPC

Malheureusement, dans certains cas, supprimer les services RPC du système n'est pas une option. Certains services de bureau local (notamment fam de SGI) sont basés sur RPC et ont donc besoin d'un portmapper local. Cela veut dire que dans certains circonstances, des utilisateurs installant un environnement de bureau (comme GNOME) installera également le portmapper.

Il y a différentes façons de limiter l'accès au portmapper et aux services RPC :

- bloquer l'accès aux ports utilisés par ces services avec un pare-feu local (consultez la section intitulée « Ajouter des capacités au pare-feu ») ;
- bloquer l'accès à ces services en utilisant l'encapsulation TCP, car le portmapper (et certains services RPC) sont compilés avec `libwrap` (consultez la section intitulée « Utilisation de tcpwrappers »). Cela veut dire que vous pouvez en bloquer l'accès par la configuration des fichiers `hosts.allow` et `hosts.deny` de l'encapsulation TCP ;

- depuis la version 5-5, le paquet portmap peut être configuré pour n'écouter que sur l'interface loopback. Pour faire cela, modifiez `/etc/default/portmap`, décommentez la ligne suivante : `#OPTIONS="-i 127.0.0.1"` et redémarrez le portmapper. Cela est suffisant pour autorisez les services locaux et en même temps pour prévenir les systèmes distants à y accéder (consultez, cependant, la section intitulée « Désactiver les problèmes d'hôtes weak-end »).

Ajouter des capacités au pare-feu

Le système d'exploitation Debian GNU/Linux possède les capacités intégrées fournies par le noyau Linux [8]. Si vous installez une version récente de Debian (le noyau installé par défaut est le 2.6) vous aurez la fonctionnalité pare-feu **iptables** (netfilter) disponible[9].

Protéger le système local avec un pare-feu

Vous pouvez utiliser des règles de pare-feu comme façon de sécuriser l'accès à votre système local et, même, de limiter les connexions sortantes effectuées par celui-ci. Des règles de pare-feu peuvent être également utilisées pour protéger des processus qui ne peuvent être proprement configurés pour *ne pas* fournir certains services à certains réseaux, certaines adresses IP, etc.

Toutefois, cette étape est présentée en dernier dans ce manuel car il est *largement* préférable de ne pas dépendre exclusivement des capacités d'un pare-feu pour protéger un système donné. La sécurité dans un système est réalisée par couches, le filtrage devrait être la dernière, une fois que tous les services ont été renforcés. Vous pouvez facilement imaginer une installation dans laquelle le système est uniquement protégé par le pare-feu et que l'administrateur enlève bêtement les règles pour n'importe quelle raison (problèmes avec l'installation, exaspération, erreur humaine, etc.), ce système pourrait être grand ouvert à une attaque s'il n'y avait aucun autre renforcement dans le système pour le protéger.

D'un autre côté, avoir des règles de pare-feu sur le système local prévient également quelques mauvaises choses de se produire. Même si les services fournis sont configurés avec sécurité, un pare-feu peut protéger des erreurs de configuration ou des services fraîchement installés qui n'ont pas encore été configurés correctement. Une configuration serrée préviendra également un cheval de Troie *appelant à la maison* de fonctionner sauf si le code de pare-feu est enlevé. Notez qu'un intrus n'a *pas* besoin de l'accès superutilisateur pour installer un cheval de Troie qui pourrait être contrôlé à distance (car l'ouverture sur des ports est autorisée si le port n'est pas privilégié et si des capacités n'ont pas été supprimées).

Une configuration correcte de pare-feu serait donc une règle de refus par défaut, c'est-à-dire :

- les connexions entrantes ne sont autorisés que pour des services locaux par des machines autorisées ;
- les connexions sortantes ne sont autorisés que pour les services utilisés par votre système (DNS, navigation web, POP, courrier, etc.)[10] ;
- la règle forward interdit tout (à moins que vous ne protégiez d'autres systèmes, voir ci-dessous) ;
- toutes les autres connexions entrantes et sortantes sont interdites.

[8]

[9] Disponible depuis le noyau 2.4 (qui était le noyau par défaut de Debian 3.0). Les versions de noyau précédentes (2.2, disponibles dans les versions encore plus anciennes de Debian) utilisaient **ipchains**. La principale différence entre **ipchains** et **iptables** est que ce dernier est basé sur une *inspection des paquets en fonction de l'état* (stateful packet inspection) qui fournit des configurations de filtrage plus sécurisées (et plus faciles à construire). Les distributions Debian plus anciennes (qui ne sont plus prises en charge) utilisant un noyau 2.0 ont besoin du correctif de noyau correspondant.

[10] À la différence des pare-feu personnels d'autres systèmes d'exploitation, Debian GNU/Linux ne fournit pas (encore) d'interface de génération de pare-feu qui puisse créer des règles les limitant par processus ou par utilisateur. Cependant, le code iptables peut être configuré pour faire cela (consultez le module propriétaire (owner) dans la page de manuel iptables(8)).

Utiliser un pare-feu pour protéger d'autres systèmes

Un pare-feu Debian peut aussi être installé de façon à protéger, selon des règles de filtrage, l'accès aux systèmes *derrière* lui, limitant leur exposition à Internet. Un pare-feu peut être configuré pour interdire l'accès de systèmes en dehors de votre réseau local à des services internes (ports) qui ne sont pas publics. Par exemple, sur un serveur de messagerie, seul le port 25 (où le service de courrier est fourni) doit être accessible depuis l'extérieur. Un pare-feu peut être configuré pour, même s'il y a d'autres services en plus des services publics, rejeter les paquets (c'est connu sous le nom de*filtrage*) dirigés vers eux.

Vous pouvez même installer une machine Debian GNU/Linux en tant que pont pare-feu, c'est-à-dire un pare-feu filtrant complètement transparent pour le réseau qui est dépourvu d'adresse IP et donc ne peut pas être attaqué directement. Selon le noyau que vous avez installé, vous pouvez avoir besoin d'installer le correctif pare-feu pour pont, puis aller à *802.1d Ethernet Bridging* lors de la configuration du noyau et une nouvelle option *netfilter (firewalling) support*. Consultez la section intitulée « Configuration d'un pare-feu pont » pour plus d'informations sur la façon de faire cela dans un système Debian GNU/Linux.

Mettre en place un pare-feu

L'installation Debian par défaut, à la différence d'autres distributions Linux, ne fournit pas encore de moyen pour l'administrateur de mettre une configuration de pare-feu lors de l'installation, mais vous pouvez installer un certain nombre de paquets de configuration de pare-feu (consultez la section intitulée « Paquets pare-feu »).

Bien sûr, la configuration du pare-feu dépend toujours du système et du réseau. Un administrateur doit connaître auparavant quelle est la disposition du réseau, les systèmes qu'il désire protéger et si d'autres considérations réseau (comme le NAT ou le routage) doivent être prises en compte ou non. Soyez prudent quand vous configurez votre pare-feu, comme le dit Laurence J. Lane dans son paquet iptables :

Les outils peuvent facilement être mal utilisés, entraînant d'énormes quantités de maux en paralysant complètement l'accès au réseau pour un système d'ordinateur. Il n'est pas très inhabituel pour un administrateur système de se bloquer lui-même en dehors du système situé à quelques centaines ou milliers de kilomètres de là. Il est même possible de se bloquer en dehors d'un ordinateur dont le clavier est sous ses doigts. Veuillez s'il vous plaît l'utiliser avec précaution.

Rappelez-vous de cela : installer simplement le paquet iptables (ou l'ancien code de pare-feu) ne vous fournit pas de protection, mais seulement les logiciels. Pour avoir un pare-feu, vous devez le *configurer* !

Si vous ne savez pas comment configurer les règles de votre pare-feu manuellement, veuillez consulter le *Packet Filtering HOWTO* et le *NAT HOWTO* fournis par iptables pour une lecture hors ligne à `/usr/share/doc/iptables/html/`.

Si vous ne connaissez pas grand chose sur les pare-feu, vous devriez commencer par lire le http://www.tldp.org/HOWTO/Firewall-HOWTO.html, installez le paquet doc-linux-text si vous voulez le lire hors ligne. Si vous désirez poser des questions ou demander de l'aide pour configurer un pare-feu, vous pouvez utiliser la liste de diffusion debian-firewall, consultez http://lists.debian.org/debian-firewall. Consultez également la section intitulée « Connaissances requises » pour plus de pointeurs (généraux) sur les pare-feu. Un autre bon tutoriel d'iptables est http://iptables-tutorial.frozentux.net/iptables-tutorial.html.

Paquets pare-feu

Configurer manuellement un pare-feu peut être compliqué pour un administrateur débutant (et même parfois pour un expert). Cependant, la communauté des logiciels libres a créé un certain nombre d'outils pouvant être utilisés pour configurer facilement un pare-feu local. Soyez prévenu que certains de ces outils

sont plus orientés vers de la protection locale seulement (également connu sous le nom de *pare-feu personnel*) et d'autres sont plus versatiles et peuvent être utilisés pour configurer des règles complexes pour protéger des réseaux entiers.

Plusieurs logiciels peuvent être utilisés pour configurer des règles de pare-feu dans un système Debian.

- Pour les systèmes de bureau :
  - firestarter, une application GNOME orientée vers les utilisateurs finaux et incluant un assistant utile pour définir rapidement des règles de pare-feu. L'application inclut une interface utilisateur pour pouvoir surveiller quand une règle de pare-feu bloque le trafic ;
  - guarddog, un paquet de configuration de pare-feu basé sur KDE orienté à la fois vers les utilisateurs novices et avancés ;
  - knetfilter, une interface graphique KDE pour gérer un pare-feu et des règles NAT pour iptables (alternative ou concurrent à l'outil guarddog bien que légèrement plus orienté vers les utilisateurs avancés) ;
  - fireflier, un outil interactif pour créer des règles iptables à partir du trafic vu sur le système et les applications. Il possède un modèle client serveur donc vous devez installer à la fois le serveur (fireflier-server) et un des clients disponibles, avec un client disponible pour chaque environnement de bureau : fireflier-client-gtk (client GTK+), fireflier-client-kde (client KDE) et fireflier-client-qt (client QT).
- Pour les systèmes serveurs (sans interface graphique) :
  - fwbuilder, une interface graphique orientée objet qui inclut des compilateurs de règles pour diverses plates-formes de pare-feu incluant netfilter de Linux, pf de BSD (utilisé dans OpenBSD, NetBSD, FreeBSD et Mac OS X) ainsi que des listes d'accès du routeur. La fonctionnalité de fwbuilder complète est également disponible depuis la ligne de commande ;
  - shorewall, un outil de configuration de pare-feu qui fournit une prise en charge IPsec ainsi qu'une prise en charge limitée pour le dimensionnement du trafic (« traffic shaping ») et la définition des règles du pare-feu. La configuration est effectuée par un simple jeu de fichiers qui sont utilisés pour générer les règles iptables ;
  - bastille, l'application de durcissement est décrit dans Chapitre 6, *Sécurisation automatique d'un système Debian*. L'une des étapes de durcissement que l'administrateur peut configurer est une définition du trafic autorisé et interdit qui est utilisée pour générer un ensemble de règles de pare-feu que le système exécutera au démarrage.

De nombreuses autres interfaces à iptables sont disponibles dans Debian ; une liste exhaustive de comparaison des différents paquets dans Debian est tenue à jour sur la http://wiki.debian.org/Firewalls.

Remarquez que certains des paquets cités ci-dessus introduiront probablement des scripts de pare-feu à exécuter lors de l'amorçage du système. Testez-les de manière exhaustive avant de redémarrer le système ou vous pourriez vous retrouver bloqué en dehors de la machine. Si vous mélangez différents paquets de pare-feu, vous pouvez obtenir des effets indésirables. Habituellement, le script de pare-feu qui s'exécute en dernier sera celui qui configurera le système (qui peut ne pas être ce que vous voulez). Consultez la documentation du paquet et utilisez l'un d'entre eux pour ces configurations.

Comme mentionné précédemment, certains programmes comme firestarter, guarddog ou knetfilter sont des interfaces graphiques pour l'administration qui utilisent soit GNOME, soit KDE (les deux derniers). Ces applications sont plus orientées utilisateur (c'est-à-dire utilisation « familiale ») tandis que certains des autres paquets de la liste sont plus orientés administrateur. Certains des programmes mentionnés auparavant (comme **bastille**) sont ciblés sur la mise en place de règles de pare-feu qui protègent l'hôte sur lequel

ils fonctionnent, mais ils ne sont pas nécessairement conçus pour mettre en place des règles de pare-feu pour des hôtes de pare-feu qui protègent un réseau (comme **shorewall** ou **fwbuilder**).

Il existe encore un autre type d'application de pare-feu: les serveurs mandataires (*proxy*) applicatifs. Si vous cherchez à mettre en place un tel pare-feu de niveau d'entreprise qui effectue du filtrage de paquets et fournit un certain nombre de serveurs mandataires transparents qui peuvent faire une analyse fine du trafic, vous devriez considérer l'utilisation de zorp, qui fournit cela dans un seul programme. Vous pouvez également mettre en place ce type de pare-feu manuellement en utilisant les serveurs mandataires disponibles dans Debian pour différents services comme pour le DNS en utilisant bind (correctement configuré), dnsmasq, pdnsd ou totd pour le FTP en utilisant frox ou ftp-proxy, pour X11 en utilisant xfwp, pour IMAP en utilisant imapproxy, pour le courrier en utilisant smtpd, ou pour POP3 en utilisant p3scan. Pour d'autres protocoles, vous devriez soit utiliser un serveur mandataire TCP générique comme simpleproxy, soit un serveur mandataire SOCKS comme dante-server, tsocks ou socks4-server. Vous devrez également typiquement utiliser un système de cache web (comme squid) et un système de filtrage web (comme squidguard ou dansguardian).

Configuration manuelle init.d

Une autre possibilité est de configurer manuellement vos règles de pare-feu par un script init.d qui exécutera toutes les commandes **iptables**. Suivez les étapes ci-dessous.

- Consultez le script ci-dessous et adaptez-le à vos besoins.
- Testez le script et vérifiez les messages du journal système pour voir le trafic qui est rejeté. Si vous testez depuis le réseau, vous voudrez soit exécuter le script shell en exemple qui enlève le pare-feu (si vous ne tapez rien pendant 20 secondes), soit commenter les définitions de règle *default deny* (*-P INPUT DROP* et *-P OUTPUT DROP*) et vérifier que le système ne rejette pas de trafic légitime.
- Déplacez le script dans `/etc/init.d/parefeu`.
- Le script ci-dessous tire avantage de l'utilisation (depuis Squeeze) par Debian d'un séquençage d'amorçage basé sur les dépendances. Pour plus d'informations voir : https://wiki.debian.org/LSBInitScripts/DependencyBasedBoot et https://wiki.debian.org/LSBInitScripts. Si les en-têtes LSB sont définis comme ils le sont dans le script, insserv configurera automatiquement le système pour qu'il lance le pare-feu avant le démarrage de n'importe quel réseau, et ne l'arrête qu'après la fermeture de tous les réseaux.

```
# insserv myfirewall
```

Voici l'exemple de script de pare-feu :

```
#!/bin/sh
# Exemple de configuration de pare-feu.
#
# Mises en garde
# - Cette configuration s'applique à toutes les interfaces réseau.
#   Si vous voulez ne restreindre cela qu'à une interface donnée,
#   utilisez « -i INTERFACE » dans les appels iptables ;
# - L'accès à distance pour les services TCP/UDP est accordé à tout
#   hôte, vous voudrez probablement restreindre cela en utilisant
#   « --source ».
#
# chkconfig : 2345 9 91
# description : activer ou désactiver le pare-feu au démarrage
#
```

```
# Vous pouvez tester ce script avant de l'appliquer avec l'extrait
# de script shell suivant, si vous ne tapez rien pendant
# 20 secondes, les règles de pare-feu seront effacées.
#-----------------------------------------------------------------
#  while true; do test=""; read  -t 20 -p "OK ? " test ; \
#  [ -z "$test" ] && /etc/init.d/parefeu clear ; done
#-----------------------------------------------------------------

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

# Services que le système offrira au réseau
TCP_SERVICES="22" # seulement SSH
UDP_SERVICES=""
# Services que le système utilisera du réseau
REMOTE_TCP_SERVICES="80" # navigation web
REMOTE_UDP_SERVICES="53" # DNS
# Réseau qui sera utilisé pour la gestion à distance
# (si non défini, aucune règle ne sera mise en place)
# NETWORK_MGMT=192.168.0.0/24
# Port utilisé pour le service SSH, à définir si vous avez configuré
# une gestion de réseau mais l'avez enlevé de TCP_SERVICES
# SSH_PORT="22"

if ! [ -x /sbin/iptables ]; then
    exit 0
fi

fw_start () {

  # Trafic d'entrée :
  /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 ${PORT} -j ACCEPT
  done
  fi
  # Gestion à distance
  if [ -n "$NETWORK_MGMT" ] ; then
    /sbin/iptables -A INPUT -p tcp --src ${NETWORK_MGMT} --dport ${SSH_PORT} -j AC
  else
    /sbin/iptables -A INPUT -p tcp --dport ${SSH_PORT}  -j ACCEPT
  fi
  # Test à distance
  /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
```

```
  # Sortie :
  /sbin/iptables -A OUTPUT -j ACCEPT -o lo
  /sbin/iptables -A OUTPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
  # ICMP est permis :
  /sbin/iptables -A OUTPUT -p icmp -j ACCEPT
  # Ainsi que les mises à jour de sécurité :
  # Remarque : vous pouvez indiquer en dur l'adresse IP ici afin de prévenir une
  # usurpation DNS et configurer les règles même si le DNS ne fonctionne pas mais
  # dans ce cas vous ne « verrez » pas les modifications d'IP pour ce service :
  /sbin/iptables -A OUTPUT -p tcp -d security.debian.org --dport 80 -j ACCEPT
  # Ainsi que pour tous les services définis :
  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
  # Toutes les autres connexions sont enregistrées dans syslog
  /sbin/iptables -A OUTPUT -j LOG
  /sbin/iptables -A OUTPUT -j REJECT
  /sbin/iptables -P OUTPUT DROP
  # Autres protections réseau
  # (certaines ne fonctionneront que pour certaines versions de noyau)
  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 "Démarrage du pare-feu…"
    fw_stop
    fw_start
    echo "done."
    ;;
  stop)
    echo -n "Arrêt du pare-feu…"
    fw_stop
    echo "done."
    ;;
  clear)
    echo -n "Effacement des règles de pare-feu…"
    fw_clear
    echo "done."
    ;;
  *)
    echo "Utilisation : $0 {start|stop|restart|clear}"
    exit 1
    ;;
  esac
exit 0
```

Au lieu d'intégrer toutes les règles iptables dans un script init.d, vous pouvez utiliser le programme **iptables-restore** pour restaurer les règles sauvées avec **iptables-save**. Pour faire cela, vous devez configurer les règles et sauver le jeu de règles dans un endroit statique (comme `/etc/default/firewall`).

Configurer les règles du réseau par ifup

Vous pouvez également utiliser la configuration du réseau dans `/etc/network/interfaces` pour mettre en place les règles de pare-feu. Pour cela, vous devez :

- créer le jeu de règles de pare-feu à appliquer quand l'interface sera active ;
- sauver le jeu de règles avec **iptables-save** dans un fichier de `/etc`, par exemple `/etc/iptables.up.rules` ;
- configurer `/etc/network/interfaces` pour utiliser le jeu de règles configurées :

```
iface eth0 inet static
        address x.x.x.x
        [… configuration de l'interface …]
        pre-up iptables-restore < /etc/iptables.up.rules
```

Optionnellement, vous pouvez mettre en place un jeu de règles à appliquer quand l'interface est *inactivée* en créant un jeu de règles, en le sauvant dans `/etc/iptables.down.rules` et en ajoutant la directive suivante à la configuration de l'interface :

```
post-down iptables-restore < /etc/iptables.down.rules
```

Pour des scripts de configuration de pare-feu plus avancés avec ifupdown, vous pouvez utiliser les accroches (*hooks*) disponibles pour chaque interface dans les répertoires `*.d/` appelés avec **run-parts** (consultez run-parts(8)).

Tester la configuration de pare-feu

Tester la configuration de pare-feu est aussi facile et aussi dangereux que d'exécuter simplement le script de pare-feu (ou d'activer la configuration que vous avez définie dans l'application de configuration de pare-feu). Cependant, si vous n'êtes pas assez prudent et que vous configurez le pare-feu à distance (comme à travers une connexion SSH), vous pourriez vous enfermer dehors.

Plusieurs moyens permettent d'empêcher cela. L'un est d'exécuter un script dans un terminal séparé qui va enlever la configuration de pare-feu si vous ne faites pas d'entrée clavier. Un exemple de cela est :

```
$  while true; do test=""; read  -t 20 -p "OK? " test ; \
  [ -z "$test" ] && /etc/init.d/firewall clear ; done
```

Un autre moyen est d'introduire une porte dérobée dans le système par un mécanisme alternatif qui vous permet soit d'enlever le système de pare-feu, soit de percer un trou dedans si quelque chose déraille. Pour cela, vous pouvez utiliser knockd et le configurer pour qu'une tentative de connexion sur un certain port enlève le pare-feu (ou ajoute une règle temporaire). Bien que les paquets soient rejetés par le pare-feu, comme **knockd** se lie à l'interface et les *voit*, vous pourrez contourner le problème.

Tester un pare-feu qui protège un réseau interne est un problème différent, vous voudrez étudier certains des outils utilisés pour le test de failles à distance (consultez la section intitulée « Outils d'évaluation des vulnérabilités à distance ») pour sonder le réseau depuis l'extérieur (ou dans toute autre direction) pour tester l'efficacité de la configuration du pare-feu.

Chapitre 6. Sécurisation automatique d'un système Debian

Après la lecture de toutes les informations des précédents chapitres, vous vous demanderez probablement : « Il y a de nombreuses choses à faire afin de sécuriser mon système, mais tout cela ne peut-il pas être automatisé ? » La réponse est oui, mais soyez prudent avec les outils automatisés. Certaines personnes pensent qu'un outil de renforcement n'élimine pas la nécessité d'une bonne administration. Donc, ne pensez pas que vous pouvez automatiser toutes les procédures et que vous arriverez à résoudre tous les problèmes. La sécurité est un processus évoluant constamment dans lequel l'administrateur doit participer et ne peut pas rester à l'écart et laisser les outils se débrouiller tout seul avec toutes les implémentations des politiques de sécurité, toutes les attaques et tous les environnements.

Depuis Woody (Debian 3.0), il existe deux paquets spécifiques qui sont utiles pour le durcissement de la sécurité. Le paquet harden qui base son approche sur les dépendances des paquets pour installer rapidement des paquets sûrs et retirer ceux avec des imperfections, la configuration devant être faite par l'administrateur. Le paquet bastille implémente une politique de sécurité donnée pour le système basée sur une configuration antérieure de l'administrateur (la configuration peut être faite à l'aide de simples questions à répondre par oui ou non).

Harden

Le paquet harden essaie de rendre plus facile l'installation et l'administration d'hôtes qui ont besoin d'une bonne sécurité. Ce paquet devrait être utilisé par ceux qui veulent une aide rapide afin d'améliorer la sécurité de leur système. Il installe automatiquement des outils pour accroître la sécurité : outils de détection d'intrusions, outils d'analyse de sécurité, etc. Harden installe les paquets *virtuels* suivants (c'est-à-dire, pas de contenu, juste des dépendances ou des recommandations vers d'autres paquets) :

- harden-tools : outils pour améliorer la sécurité du système (vérificateur d'intégrité, détection d'intrusions, correctifs pour noyau, etc.) ;
- harden-environment : aide à configurer un durcissement d'environnement (actuellement vide) ;
- harden-servers : retire les serveurs considérés comme douteux pour certaines raisons ;
- harden-clients : retire les clients considérés comme douteux pour certaines raisons ;
- harden-remoteaudit : outils pour auditer un système à distance ;
- harden-nids : outils pour installer un système de détection d'intrusions ;
- harden-surveillance : outils pour surveiller les réseaux et les services.

Paquets utiles qui ne sont pas une dépendance :

- harden-doc : fournit ce même manuel et d'autres paquets de documentation liés à la sécurité ;
- harden-development : outils de développement pour créer des programmes plus sécurisés.

Prenez garde dans le cas où vous avez besoin d'un logiciel (et que vous ne voulez pas désinstaller) et qu'il soit en contradiction avec certains paquets ci-dessus, vous ne serez peut-être pas capable d'utiliser pleinement harden. Les paquets *harden* ne font rien directement. Cependant, ils entrent en conflit avec des paquets reconnus comme étant risqués. De cette façon, le système de paquets de Debian n'approuvera pas

automatiquement l'installation de ces paquets. Par exemple, si vous tentez d'installer un serveur TELNET alors que harden-servers est installé, apt vous dira :

```
# apt-get install telnetd
Les paquets suivants seront ENLEVÉS :
  harden-servers
Les NOUVEAUX paquets suivants seront installés :
  telnetd
Souhaitez-vous continuer ? [O/n]
```

Cela devrait alerter l'administrateur, qui devrait reconsidérer ses actions.

Bastille Linux

htttp://bastille-linux.sourceforge.net/ est un outil de durcissement automatique originellement orienté vers les distributions Linux Red Hat et Mandrake. Toutefois, le paquet bastille fourni dans Debian (depuis Woody) a été modifié de façon à fournir les mêmes fonctionnalités pour les systèmes Debian GNU/Linux.

Bastille peut être utilisé avec différentes interfaces (toutes sont documentées dans leur propre page de manuel dans le paquet Debian) qui permettent à l'administrateur de :

- Répondre aux questions, étape par étape, concernant le niveau de sécurité désiré de votre système (en utilisant InteractuveBastille(8)).
- utiliser un paramétrage par défaut pour la sécurité (parmi trois : relachée, modérée ou paranoïaque) dans une installation définie (serveur ou poste de travail) et laisser Bastille décider quelle politique de sécurité appliquer (en utilisant BastilleChooser(8)) ;
- prendre un fichier de configuration prédéfini (qui peut être fourni par Bastille ou créé par l'administrateur) et implémenter une politique de sécurité donnée (en utilisant AutomatedBastille(8)).

Chapitre 7. Infrastructure de sécurité Debian

L'équipe de sécurité Debian

Debian possède une équipe de sécurité, qui assure la sécurité dans la distribution *stable*. Assurer la sécurité veut dire suivre les failles qui surviennent dans les logiciels (en surveillant des forums comme Bugtraq ou vuln-dev) et déterminer si la distribution *stable* est concernée par ces failles.

L'équipe de sécurité Debian est également le point de contact pour les problèmes qui sont coordonnés par les développeurs amont ou des organisations comme le htttp://www.cert.org, qui peuvent toucher de multiples distributeurs, c'est-à-dire quand les problèmes ne sont pas spécifiques à Debian. Le point de contact avec l'équipe de sécurité est mailto:team@security.debian.org qui n'est lu que par les membres de l'équipe de sécurité.

Les informations secrètes devraient être envoyées à la première adresse et, dans certains cas, devraient être chiffrées avec la clef du contact de l'équipe de sécurité (disponible dans le trousseau Debian).

Dès qu'un problème probable est reçu par l'équipe de sécurité, elle recherchera si la distribution *stable* est affectée et si c'est le cas, un correctif sera créé pour la base de code source. Ce correctif contiendra parfois un rétroportage du correctif effectué en amont (qui est habituellement en avance de plusieurs versions par rapport à la version distribuée par Debian). Après qu'un test du correctif ait été effectué, les nouveaux paquets sont préparés et publiés sur le site htttp://security.debian.org pour pouvoir être récupérés par **apt** (consultez la section intitulée « Faire une mise à jour de sécurité »). En même temps, une *alerte de sécurité Debian* (Debian Security Advisory ou DSA) est publiée sur le site web et envoyée aux listes de diffusion publiques y compris htttp://lists.debian.org/debian-security-announce et Bugtraq.

D'autres questions souvent posées sur l'équipe de sécurité Debian peuvent être trouvées en la section intitulée « Questions concernant l'équipe de sécurité Debian ».

Alertes de sécurité Debian

Les alertes de sécurité Debian (DSA) sont effectuées à chaque fois qu'une faille affectant un paquet Debian est découverte. Ces alertes, signées par l'un des membres de l'équipe de sécurité, contiennent des renseignements sur les versions touchées ainsi que l'emplacement des mises à jour. Ces informations sont :

- numéro de version pour le correctif ;
- type de problème ;
- s'il est exploitable à distance ou localement ;
- description courte du paquet ;
- description du problème ;
- description du stratagème ;
- description du correctif.

Les DSA sont publiées sur htttp://www.debian.org/ et dans les htttp://www.debian.org/security/. Cela ne se produit habituellement pas avant que le site web ne soit reconstruit (toutes les quatre heures), elles

peuvent donc ne pas être immédiatement présentes. Le canal préféré est la liste de diffusion debian-security-announce.

Les utilisateurs intéressés peuvent, cependant (et c'est fait sur quelques portails relatifs à Debian) utiliser le flux RDF pour télécharger automatiquement les DSA sur leur bureau. Certaines applications, comme **Evolution** (un client de messagerie et assistant d'informations personnelles) et **Multiticker** (une applette GNOME) peuvent être utilisées pour récupérer les alertes automatiquement. Le flux RDF est disponible à htttp://www.debian.org/security/dsa.rdf.

Les DSA publiées sur le site web peuvent être mises à jour après avoir été envoyées sur les listes de diffusion publiques. Une mise à jour courante est d'ajouter des références croisées vers les bases de données des failles de sécurité. Les traductions[1] des DSA ne sont pas envoyées aux listes de diffusion de sécurité, mais elles sont directement intégrées au site web.

Références croisées des failles

Debian fournit une http://www.debian.org/security/crossreferences complète comprenant toutes les références disponibles pour toutes les alertes publiées depuis 1998. Cette table est fournie en complément de la http://cve.mitre.org/cve/refs/refmap/source-DEBIAN.html.

Vous remarquerez que cette table fournit des références vers des bases de données de sécurité comme htttp://www.securityfocus.com/bid, les htttp://www.cert.org/advisories/ et la htttp://www.kb.cert.org/vuls ainsi que les noms CVE (voir ci-dessous). Ces références sont fournies pour faciliter l'utilisation, mais seules les références CVE sont périodiquement vérifiées et intégrées.

Les avantages d'ajouter les références croisées vers ces bases de données de failles sont que :

- cela permet plus facilement aux utilisateurs de Debian de voir et de suivre quelles alertes générales (publiées) ont déjà été couvertes par Debian ;
- les administrateurs système peuvent en apprendre plus sur la faille et ses impacts en suivant les références croisées ;
- ces renseignements peuvent être utilisés pour vérifier les sorties de scanneurs de failles qui contiennent des références à CVE pour supprimer des faux positifs (consultez la section intitulée « Le scanneur X de vérification des failles indique que le système Debian est vulnérable ! »).

Compatibilité CVE

Les alertes de sécurité Debian ont été http://www.debian.org/security/CVE-certificate.jpg[2] le 24 février 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.

Le projet htttp://cve.mitre.org est maintenu par la société MITRE et fournit une liste des noms standardisés pour les failles et expositions de sécurité.

Debian estime que fournir aux utilisateurs des informations supplémentaires liées aux problèmes de sécurité qui touchent la distribution Debian est extrêmement important. L'inclusion des noms CVE dans les

[1] Des traductions sont disponibles jusqu'en dix langues.
[2] Le http://cve.mitre.org/compatible/phase2/SPI_Debian.html complet est disponible au CVE.

alertes aide les utilisateurs à associer des failles génériques avec les mises à jour spécifiques de Debian, ce qui réduit le temps passé à gérer les failles qui concernent nos utilisateurs. Cela facilite également la gestion du risque dans un environnement où sont déployés des outils de sécurité gérant CVE — comme des systèmes de détection d'intrusion d'hôte ou de réseau ou des outils de vérification de failles — qu'ils soient ou non basés sur la distribution Debian.

Debian fournit maintenant les noms CVE pour toutes les DSA publiées depuis septembre 1998. Toutes les alertes peuvent être récupérées sur le site web Debian et les annonces liées aux nouvelles failles contiennent les noms CVS quand ils sont disponibles lors de leur publication. Les alertes liées à un nom CVE donné peuvent être cherchées directement avec le système de suivi en sécurité Debian (voir ci-après).

Dans certains cas, vous pouvez ne pas trouver un nom CVE donné dans les alertes publiées par exemple parce que :

- aucun produit Debian n'est concerné par cette faille ;
- il n'y a pas encore eu d'alerte couvrant cette faille (le problème de sécurité peut avoir été signalé comme un htttp://bugs.debian.org/cgi-bin/pkgreport.cgi?tag=security, mais aucune correction n'a encore été testée et envoyée) ;
- une alerte a été publiée avant qu'un nom CVE ait été attribué à une faille donnée (chercher une mise à jour sur le site web).

Système de suivi en sécurité

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.

Le paquet **debsecan** utilise les renseignements du système de suivi pour signaler à l'administrateur d'un système les paquets installés vulnérables, et ceux pour lesquels des mises à jour corrigeant les problèmes de sécurité sont disponibles.

Infrastructure de construction de sécurité Debian

Comme Debian prend actuellement en charge un grand nombre d'architectures, les administrateurs se demandent parfois si une architecture donnée pourrait prendre plus de temps pour recevoir des mises à jour de sécurité qu'une autre. En fait, à part dans de rares circonstances, les mises à jour sont disponibles pour toutes les architectures en même temps.

Les paquets de l'archive de sécurité sont construits automatiquement, tout comme l'archive classique. Cependant, les mises à jour de sécurité sont un petit peu différentes des envois normaux par les responsables de paquets car, dans certains cas, avant d'être publiées, elles doivent attendre de pouvoir être plus testées, qu'une alerte soit rédigée ou attendre une semaine ou plus pour éviter de publier le défaut jusqu'à ce que tous les distributeurs aient eu une chance raisonnable de le corriger.

L'archive d'envoi de sécurité fonctionne donc de la façon suivante :

- quelqu'un trouve un problème de sécurité ;

- quelqu'un corrige le problème et fait un envoi vers *incoming* de security-master.debian.org (ce *quelqu'un* est habituellement un membre de l'équipe de sécurité, mais ce peut aussi être un responsable de paquet avec un correctif approprié qui a contacté l'équipe de sécurité auparavant). Le journal de modifications contient une cible de distribution *testing-security* ou *stable-security* ;

- l'envoi est vérifié et traité par un système Debian et déplacé dans queue/accepted et le service d'empaquetage est prévenu. Les fichiers à cet endroit sont accessibles par l'équipe de sécurité et (de façon un peu indirecte) par les service d'empaquetage ;

- les serveurs d'empaquetage activés pour la sécurité récupèrent le paquet source (en priorité par rapport aux constructions courantes), le construisent et envoient les journaux à l'équipe de sécurité ;

- l'équipe de sécurité répond aux journaux et les paquets nouvellement construits sont envoyés vers queue/unchecked, où ils sont traités par un système Debian et déplacés dans queue/accepted ;

- quand l'équipe de sécurité trouve les paquets acceptables (c'est-à-dire qu'ils sont correctement construits pour toutes les architectures pertinentes et corrigent le trou de sécurité sans introduire de nouveau problème par eux-mêmes), un script est exécuté qui :

  - installe le paquet dans l'archive de sécurité ;

  - met à jour les fichiers `Packages`, `Sources` et `Release` de security.debian.org de la façon habituelle (**dpkg-scanpackages**, **dpkg-scansources**, etc.) ;

  - met en place un modèle d'alerte que l'équipe de sécurité peut compléter ;

  - fait suivre les paquets vers le proposed-updates approprié pour qu'il soit intégré à l'archive réelle dès que possible.

Cette procédure, auparavant réalisée à la main, a été testée et mise en place pendant l'étape de gel de Debian 3.0 Woody (juillet 2002). Grâce à cette architecture, l'équipe de sécurité a pu avoir des paquets mis à jour pour les problèmes d'Apache et d'OpenSSH pour toutes les architectures prises en charge (presque vingt) en moins d'un jour.

Le guide du développeur pour les mises à jour de sécurité

Les développeurs Debian qui doivent se coordonner avec l'équipe en charge de la sécurité pour corriger un problème avec leurs paquets, peuvent consulter la référence du développeur section htttp://www.debian.org/doc/manuals/developers-reference/pkgs.html#bug-security.

La signature de paquet dans Debian

Ce chapitre pourrait également être intitulé « comment mettre à jour et à niveau un système Debian GNU/Linux en sécurité » et mérite d'avoir son propre chapitre car c'est une partie importante de l'infrastructure de sécurité. La signature des paquets est un point important car elle évite l'altération de paquets distribués sur les miroirs et des téléchargements avec des attaques en homme au milieu (« man-in-the-middle »). Les mises à jour de logiciels automatiques sont une fonctionnalité importante, mais il est également important d'enlever les menaces de sécurité qui pourraient favoriser la propagation de chevaux de Troie et la compromission de systèmes lors des mises à jour[3].

[3] Certains systèmes d'exploitation ont déjà été touchés par des problèmes de mises à jour automatiques comme la htttp://www.cunap.com/~hardingr/projects/osx/exploit.html.

FIXME : la faille d'Internet Explorer sur la gestion des chaînes de certificat a probablement eu un impact sur les mises à jour de sécurité de Microsoft Windows.

Debian ne fournit pas de paquets signés, mais fournit un mécanisme disponible depuis Debian 4.0 *Etch* pour vérifier l'intégrité des paquets téléchargés[4]. Pour obtenir plus de renseignements, consultez la section intitulée « apt sécurisé ».

Ce problème est mieux décrit dans le http://www.cryptnet.net/fdp/crypto/strong_distro.html par V. Alex Brennen.

Le schéma actuel pour la vérification de paquet

Le schéma actuel pour la vérification de signatures de paquet en utilisant **apt** est :

- le fichier `Release` contient la somme de contrôle MD5 de `Packages.gz` (qui contient les sommes de contrôle MD5 des paquets) et sera signé. La signature est celle d'une source sûre ;
- ce fichier `Release` est téléchargé par « apt-get update » et stocké sur le disque dur avec `Packages.gz` ;
- quand un paquet est sur le point d'être installé, il est d'abord téléchargé, puis la somme MD5 est calculée ;
- le fichier `Release` signé est vérifié (la signature est bonne) et la somme MD5 en est extraite pour le fichier `Packages.gz`, la somme de contrôle de `Packages.gz` est calculée et (si elle est bonne) la somme de contrôle MD5 du paquet téléchargé en est extraite ;
- si la somme de contrôle MD5 du paquet téléchargé est la même que celle contenue dans le fichier `Packages.gz`, le paquet sera installé sinon l'administrateur sera averti et le paquet sera laissé dans le cache (ainsi l'administrateur décidera de l'installer ou non). Si le paquet n'est pas dans `Packages.gz` et que l'administrateur a configuré le système pour installer uniquement les paquets vérifiés, il ne sera pas installé non plus.

En suivant la chaîne des sommes MD5, **apt** est capable de vérifier qu'un paquet est originaire d'une version bien spécifique. C'est moins souple que de signer chaque paquet un par un, mais ce peut être combiné également avec ce schéma (voir ci-dessous).

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 la section intitulée « apt sécurisé ». 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**.

La signature de paquets a été abordée dans Debian depuis pas mal de temps déjà, pour plus d'informations vous pouvez lire : htttp://www.debian.org/News/weekly/2001/8/ et htttp://www.debian.org/News/weekly/2000/11/.

apt sécurisé

La version 0.6 d'apt, disponible depuis Debian 4.0 *Etch*, et les versions plus récentes, intègrent *apt-secure* (aussi connu sous le nom d'*apt sécurisé*) qui est un outil permettant à l'administrateur système de tester l'intégrité des paquets téléchargés conformément au schéma ci-dessus. Cette version contient l'outil **apt-key** pour ajouter de nouvelles clefs au trousseau d'apt qui ne contient par défaut que la clef actuelle de signature de l'archive Debian.

[4] Les versions plus anciennes, comme Debian 3.1 *Sarge* peuvent utiliser cette fonctionnalité en utilisant les versions rétroportées de cet outil de gestion de paquets.

Ces modifications sont basées sur un correctif pour **apt** (disponible dans le htttp://bugs.debian.org/cgi-bin/bugreport.cgi?bug=203741) qui fournit cette implémentation.

apt sécurisé fonctionne en vérifiant la distribution à l'aide du fichier `Release`, conformément à la section intitulée « Vérification par version de distribution ». Typiquement, ce processus sera transparent pour l'administrateur bien qu'il faudra intervenir chaque année [5] pour ajouter la nouvelle clef de l'archive quand elle est modifiée. Pour obtenir plus de renseignements sur les étapes qu'un administrateur doit accomplir, consultez la section intitulée « Ajout de clef en sécurité ».

Cette fonctionnalité est encore en développement, donc si vous pensez avoir trouvé des bogues dans ce paquet, veuillez d'abord vérifier que vous utilisez la dernière version (car ce paquet peut évoluer beaucoup avant d'être diffusé) et si vous utilisez la dernière version, soumettez un rapport de bogue sur le paquet apt.

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/.

Vérification par version de distribution

Cette section décrit le mode de fonctionnement du mécanisme de vérification par version de distribution, elle a été écrite par Joey Hess et est également disponible dans le htttp://wiki.debian.org/SecureApt.

Concepts de base

Voici quelque concepts de base que vous devrez comprendre pour la suite de cette section.

Une somme de contrôle est une méthode permettant de prendre un fichier et de le réduire en un nombre suffisamment petit qui identifie son contenu de façon unique. C'est beaucoup plus compliqué qu'il n'y parait de faire ça bien, et le type de sommes de contrôle le plus fréquemment utilisé, MD5, est en passe d'être cassé.

La cryptographie à clef publique est basée sur une paire de clefs: une publique et une privée. La clef publique est distribuée partout ; la clef privée doit être gardée secrète. Tous ceux qui possèdent la clef publique peuvent chiffrer un message qui ne pourra être lu que par un possesseur de la clef privée. La clef privée permet elle de signer un fichier, pas de le chiffrer. Si une clef privée est utilisée pour signer un fichier, alors tous ceux qui ont la clef publique peuvent vérifier que le fichier était signé par cette clef. Une personne ne possédant pas la clef privée ne peut pas contrefaire une telle signature.

Ces clefs sont des nombres assez grands (de 1024 à 2048 chiffres, ou plus) et pour les rendre plus facile à utiliser, ils ont un identifiant de clef, plus court (un nombre de 8 ou 16 chiffres), qui peut être utilisé pour y référer.

gpg est l'outil utilisé par apt sécurisé pour signer les fichiers et vérifier leurs signatures.

apt-key est un programme qui permet de gérer un trousseau de clefs GPG pour apt sécurisé. Le trousseau est gardé dans le fichier `/etc/apt/trusted.gpg` (à ne pas confondre avec le fichier `/etc/apt/trustdb.gpg` relatif, mais pas très intéressant). **apt-key** permet de montrer les clefs du trousseau, et d'ajouter ou enlever une clef.

Sommes de contrôle de Release

Une archive Debian contient un fichier `Release`, qui est mis à jour à chaque fois qu'un paquet de l'archive est modifié. Entre autres, le fichier `Release` contient les sommes MD5 d'autres fichiers de l'archives. Exemple d'extrait de fichier `Release` :

[5] Jusqu'à ce qu'un mécanisme automatique ne soit développé.

```
MD5Sum:
 6b05b392f792ba5a436d590c129de21f            3453 Packages
 1356479a23edda7a69f24eb8d6f4a14b            1131 Packages.gz
 2a5167881adc9ad1a8864f281b1eb959            1715 Sources
 88de3533bf6e054d1799f8e49b6aed8b             658 Sources.gz
```

Les fichiers `Release` contiennent aussi des sommes de contrôle SHA-1, ce qui sera utile quand les sommes de contrôle MD5 seront complètement cassées, toutefois, apt ne les utilise pas encore.

Maintenant, à l'intérieur d'un fichier `Packages`, d'autres sommes de contrôle MD5 sont disponibles : une pour chaque paquet de la liste. Par exemple :

```
Package: uqm
Priority: optional
...
Filename: unstable/uqm_0.4.0-1_i386.deb
Size: 580558
MD5sum: 864ec6157c1eea88acfef44d0f34d219
```

Ces deux sommes de contrôle permettent de vérifier que la copie du fichier `Packages` téléchargé est correcte, avec une somme de contrôle MD5 qui correspond à celle du fichier `Release`. Lorsqu'un paquet est téléchargé individuellement, la vérification de la somme de contrôle MD5 avec le contenu du fichier `Packages` est aussi possible. Si apt échoue à l'une de ces étapes, il abandonnera.

Rien de nouveau pour apt sécurisé, mais cela fournit les bases. Remarquez qu'un seul fichier n'a pas pu être vérifié par apt : le fichier `Release`. apt sécurisé a justement pour but de faire vérifier `Release` par apt avant de faire quoi que ce soit d'autre avec, et de combler ce trou, afin de rendre la chaîne de vérification complète, du paquet qui va être installé jusqu'au fournisseur du paquet.

Vérification du fichier Release

Pour vérifier le fichier `Release`, une signature gpg est ajoutée dans le fichier `Release.gpg`, distribué à ses côtés. Il ressemble à ceci[6], bien que seul gpg accède à son contenu normalement :

```
-----BEGIN PGP SIGNATURE-----
Version: GnuPG v1.4.1 (GNU/Linux)

iD8DBQBCqKO1nukh8wJbxY8RAsfHAJ9hu8oGNRAl2MSmP5+z2RZb6FJ8kACfWvEx
UBGPVc7jbHHsg78EhMBlV/U=
=x6og
-----END PGP SIGNATURE-----
```

Vérification du fichier Release.gpg par apt

apt sécurisé télécharge les fichiers `Release.gpg` en même temps que les fichiers `Release`, et s'il ne peut pas télécharger `Release.gpg`, ou si la signature n'est pas correcte, il se plaindra, et fera remarquer que les fichiers `Packages` pointés par le fichier `Release`, et tous les paquets contenus dedans, ne proviennent pas d'une source de confiance. Voici à quoi cela ressemble lors d'un **apt-get update** :

```
W: Erreur de GPG : http://ftp.us.debian.org testing Release :
```

[6] D'un point de vue technique, c'est une signature ASCII-armored détachée.

```
Les signatures suivantes n'ont pas pu être vérifiées car la
clé publique n'est pas disponible : NO_PUBKEY 010908312D230C5F
```

Remarquez que la seconde partie du grand nombre est l'identifiant de la clef qu'apt ne connaît pas, c'est-à-dire 2D230C5F ici.

Si vous ignorez cet avertissement et essayez d'installer un paquet ensuite, apt avertira de nouveau :

```
ATTENTION : les paquets suivants n'ont pas été authentifiés.
  libglib-perl libgtk2-perl
Faut-il installer ces paquets sans vérification (o/N) ?
```

Si vous acceptez ici, vous n'avez aucun moyen de savoir si le fichier que vous obtenez est le paquet que vous voulez installer, ou s'il s'agit d'autre chose que quelqu'un pouvant intercepter la communication avec le serveur[7] a préparé pour vous, avec une mauvaise surprise.

Remarquez que vous pouvez désactiver ces vérifications en exécutant apt avec --allow-unauthenticated.

Remarquez également que les nouvelles versions de l'installateur Debian utilisent le même mécanisme de fichier `Release` lors du debootstrap du système de base Debian, avant qu'apt ne soit disponible, et que l'installateur utilise même ce système pour vérifier ses morceaux qu'il télécharge. Enfin, Debian ne signe pour l'instant pas les fichiers `Release` de ses CD ; apt peut être configuré pour faire toujours confiance aux fichiers des CD de sorte que ce ne soit pas un gros problème.

Comment expliquer à apt en quoi avoir confiance

La sécurité de l'intégralité du système dépend de l'existence d'un fichier `Release.gpg`, qui signe un fichier `Release` et de la vérification d'**apt** à l'aide de gpg. Pour vérifier la signature, il doit connaître la clef publique de la personne qui a signé le fichier. Ces clefs sont gardées dans le trousseau spécifique à apt (`/etc/apt/trusted.gpg`) et apt sécurisé arrive avec la gestion des clefs.

Par défaut, les systèmes Debian sont fournis préconfigurés avec la clef d'archive Debian du trousseau.

```
# apt-key list
/etc/apt/trusted.gpg
--------------------
pub   1024D/4F368D5D 2005-01-31 [expire: 2006-01-31]
uid                  Debian Archive Automatic Signing Key (2005) <ftpmaster@debian
```

Ici 4F368D5D est l'identifiant de clef, et remarquez que la clef n'est valable que pour une période d'un an. Debian permute ces clefs comme dernière ligne de défense contre une quelconque brèche de sécurité de cassage de clef.

apt aura ainsi confiance en l'archive Debian officielle, mais si vous ajoutez d'autres dépôts apt à `/etc/apt/sources.list`, il vous faudra également donner à **apt** sa clef si vous voulez qu'il ait confiance en ce dépôt. Une fois que vous possédez la clef et que vous l'avez vérifiée, il suffit d'exécuter **apt-key add fichier** pour l'ajouter. Obtenir la clef et la vérifier sont les parties les plus délicates.

Trouver la clef d'un dépôt

Le paquet debian-archive-keyring est utilisé pour distribuer les clefs à **apt**. Les mises à niveau de ce paquet peuvent ajouter (ou retirer) des clefs gpg pour l'archive Debian principale.

[7] Ou ayant empoisonné le DNS, ou usurpant le serveur, ou ayant remplacé le fichier sur le miroir utilisé, etc.

Pour les autres archives, il n'y a pas encore d'endroit normalisé pour trouver la clef d'un dépôt apt donné. La clef est souvent liée depuis la page web du dépôt ou placée dans le dépôt directement, mais il vous faudra parfois la chercher.

The Debian archive signing key is available at https://ftp-master.debian.org/keys.html.[8]

gpg a lui même un moyen normalisé de distribuer les clefs, utilisant un servent de clefs, d'où gpg peut télécharger une clef pour l'ajouter à son trousseau. Par exemple :

```
$ gpg --keyserver pgpkeys.mit.edu --recv-key 2D230C5F
gpg: requête de la clé 2D230C5F du serveur hkp pgpkeys.mit.edu
gpg: clé 2D230C5F: clé publique « Debian Archive Automatic Signing Key (2006)
<ftpmaster@debian.org> » importée
gpg:        Quantité totale traitée: 1
gpg:                      importée: 1
```

Vous pouvez alors exporter cette clef depuis votre propre trousseau et la fournir à **apt-key** :

```
$ gpg -a --export 2D230C5F | sudo apt-key add -
gpg: aucune clé de confiance ultime n'a été trouvée
OK
```

L'avertissement « gpg: aucune clé de confiance ultime n'a été trouvée » signifie que gpg n'était pas configuré pour faire confiance de façon ultime à une clef en particulier. Les réglages de confiance font partie du réseau de confiance d'OpenPGP qui ne s'applique pas ici. Cet avertissement n'est donc pas un problème ici. Dans les configurations typiques, seule la propre clef de l'utilisateur est de confiance ultime.

Ajout de clef en sécurité

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).

Être paranoïaque est une bonne attitude en sécurité, mais vérifier les choses à partir d'ici est plus difficile. **gpg** connaît le concept de chaîne de confiance, qui peut commencer à partir de quelqu'un dont vous être sûr, qui signe la clef de quelqu'un, qui signe une autre clef, etc. jusqu'à atteindre la clef de l'archive. Si vous êtes suffisamment paranoïaque, vous voudrez vérifier que la clef de l'archive est signée par une clef en laquelle vous pouvez avoir confiance, avec une chaîne de confiance qui remonte jusqu'à quelqu'un que vous connaissez personnellement. Si vous voulez faire cela, rendez vous à une conférence Debian ou peut-être à un groupe (LUG) local pour une signature de clef [9].

Si vous ne pouvez pas vous permettre ce niveau de paranoïa, faites le nécessaire suffisant de votre point de vue quand vous ajoutez une nouvelle source apt et une nouvelle clef. Peut-être voudrez vous échanger un courrier électronique avec la personne fournissant la clef et la vérifier, ou peut-être préférerez vous tenter votre chance en téléchargeant la clef en supposant que c'est la bonne. Ce qui est important est qu'en

[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.
[9] Toutes les clefs de dépôt apt ne sont pas encore signées par une autre clef. Peut-être que la personne qui a mis en place le dépôt n'a pas d'autre clef, ou peut-être que ça ne lui plaît pas de signer une telle clef de rôle avec sa clef principale. Pour des renseignements au sujet de la mise en place d'une clef de dépôt, consultez la section intitulée « Vérification de distribution pour les sources non Debian ».

réduisant le problème au niveau de confiance des clefs de l'archive, apt sécurisé vous laisse être aussi prudent et sécurisé que vous désirez l'être.

Vérification de l'intégrité des clefs

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/keys/archive-key-6.0.asc | gpg --import
gpg: clé 473041FA: clé publique « Debian Archive Automatic Signing Key (6.0/squeez
  <ftpmaster@debian.org> » importée
gpg:        Quantité totale traitée: 1
gpg:                       importée: 1  (RSA: 1)
gpg: 3 marginale(s) nécessaires, 1 complète(s) nécessaires, modèle
  de confiance PGP
gpg: profondeur: 0  valide:   1  signé:   0
confiance: 0-. 0g. 0n. 0m. 0f. 1u
$ gpg --check-sigs --fingerprint 473041FA
pub   4096R/473041FA 2010-08-27 [expire: 2018-03-05]
    Empreinte de la clé = 9FED 2BCB DCD2 9CDF 7626  78CB AED4 B06F 4730 41FA
uid                  Debian Archive Automatic Signing Key (6.0/squeeze) <ftpmaster
sig!3        473041FA 2010-08-27  Debian Archive Automatic Signing Key (6.0/squeez
  <ftpmaster@debian.org>
sig!         7E7B8AC9 2010-08-27  Joerg Jaspert <joerg@debian.org>
sig!    P    B12525C4 2010-08-27  Joerg Jaspert <joerg@debian.org>
sig!         D0EC0723 2010-08-27  Mark Hymers <mhy@debian.org>
sig!         8AEA8FEE 2010-08-27  Stephen Gran <steve@lobefin.net>
sig!         A3AE44A4 2010-08-28  Michael O'Connor (stew) <stew@vireo.org>
sig!         00D8CD16 2010-08-28  Alexander Reichle-Schmehl <alexander@reichle.sch
sig!         CD15A883 2010-08-28  Alexander Schmehl (privat) <alexander@schmehl.in
sig!         672C8B12 2010-08-28  Alexander Reichle-Schmehl <tolimar@debian.org>
sig!2        C4CF8EC3 2010-08-28  Torsten Werner <twerner@debian.org>
sig!2        D628A5CA 2010-08-28  Torsten Werner <mail.twerner@googlemail.com>
```

and then as in la section intitulée « La signature de paquet dans 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 473041FA | sudo apt-key add -
OK
```

Remarquez que la clef est signée par la clef de l'archive précédente, donc vous pouvez en théorie vous appuyer simplement sur votre confiance précédente.

Rotation annuelle de la clef de l'archive Debian

Comme signalé précédemment, la clef de l'archive Debian est modifiée tous les ans, en janvier. Comme apt sécurisé est encore jeune, nous manquons encore d'expérience pour modifier la clef et des passages sont un peu abrupts.

En janvier 2006, une nouvelle clef à été préparée pour 2006 et le fichier `Release` a commencé à être signé par cette clef, mais pour éviter de casser les systèmes qui utilisaient encore l'ancienne clef de 2005,

le fichier `Release` était aussi signé par cette dernière. Le but était qu'apt accepte les deux signatures, indépendamment de la clef qu'il possède, mais à cause d'un bogue d'apt, il refusait de faire confiance au fichier s'il n'avait pas les deux clefs et n'était pas capable de vérifier les deux signatures. Cela a été corrigé dans la version 0.6.43.1 d'apt. Une confusion existait aussi sur la façon de distribuer la clef aux utilisateurs qui utilisaient déjà des systèmes avec apt sécurisé ; elle avait été envoyée sur le site web sans annonce et sans réel moyen de la vérifier, et les utilisateurs ont été obligés de la télécharger eux-mêmes.

En janvier 2006, une nouvelle clef à été préparée pour 2006 et le fichier `Release` a commencé à être signé par cette clef, mais pour éviter de casser les systèmes qui utilisaient encore l'ancienne clef de 2005, le fichier `Release` était aussi signé par cette dernière. Pour éviter les confusions sur le meilleur mécanisme de distribution pour les utilisateurs qui utilisent déjà des systèmes avec apt sécurisé, le paquet debian-archive-keyring a été introduit, pour gérer les mises à jour du trousseau de clefs d'apt.

Problèmes connus de vérification de la publication

Un autre problème évident est que si l'horloge est très décalée, apt sécurisé ne fonctionnera pas. Si la date est configurée dans le passée, comme en 1999, apt échouera avec un message peu compréhensible comme :

```
W: GPG error: http://ftp.us.debian.org sid Release: Unknown error executing gpg
```

Pourtant **apt-key** list expliquera le problème :

```
gpg: la clé 473041FA a été créée 367773259 secondes dans le futur (rupture
spatio-temporelle ou problème d'horloge)
pub   4096R/473041FA 2010-08-27 [expire: 2018-03-05]
uid                  Debian Archive Automatic Signing Key (6.0/squeeze) <ftpmaster
```

Si elle est configurée à une date trop dans le futur, apt considérera la clef expirée.

Un autre problème que vous pourriez rencontrer en utilisant testing ou unstable, est que si vous n'avez pas exécuté **apt-get update** récemment et **apt-get install** un paquet, apt risque de se plaindre qu'il ne peut pas être authentifié. **apt-get update** corrigera cela.

Vérification manuelle par version de distribution

Au cas où vous voudriez ajouter des vérifications de sécurité supplémentaires et que vous ne vouliez pas ou pouviez pas utiliser la dernière version d'apt[10] vous pouvez utiliser le script ci-dessous fourni par Anthony Towns. Ce script peut automatiquement faire certaines nouvelles vérifications de sécurité qui permettent à l'utilisateur d'être sûr que le logiciel qu'il télécharge correspond à celui de la distribution de logiciels Debian. Cela empêche les développeurs Debian d'intégrer des nouveautés au système de quelqu'un en outrepassant les responsabilités qui incombent au chargement vers l'archive principale, ou encore cela empêche une duplication similaire mais pas exactement identique, ou pour finir cela empêche l'utilisation de miroirs fournissant des copies anciennes de la version unstable ou connaissant des problèmes de sécurité.

Ce code exemple, renommé en **apt-release-check**, devrait être utilisé de la manière suivante :

```
# apt-get update
# apt-check-sigs
```

[10] Soit parce que vous utilisez la version stable *Sarge* ou une version plus ancienne, soit parce que vous ne voulez pas utiliser la dernière version d'apt, bien que nous apprécierions qu'elle soit testée

```
(...résultats...)
# apt-get dist-upgrade
```

Avant tout, vous avez besoin de :

- 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
```

- retirer toutes les lignes de `/etc/apt/sources.list` qui n'utilisent pas la structure normale « dists » ou modifier le script afin qu'il fonctionne avec elles ;

- être prêt à ignorer le fait que les mises à jour de sécurité Debian n'ont pas de fichiers `Release` signés et que les fichiers `Sources` n'ont pas (encore) les sommes de contrôle (« checksums ») appropriées dans le fichier `Release` ;

- être prêt à vérifier que les sources appropriées soient signées par les clefs appropriées.

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 "Succès"
                        return
                fi
                echo "$FILE" >>MISSING
                echo "$Y manquant"
                return
        fi
        if [ "$Y" = "" ]; then
                echo "$FILE" >>NOCHECK
                echo "Pas de vérification"
                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 "Problème"
                return
        fi
        echo "$FILE" >>OK
        echo "Succès"
}

echo
echo "Vérification des sources dans /etc/apt/sources.list :"
echo "~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~"
echo
(echo "Vous devriez vous assurer que les distributions que vous téléchargez"
echo "sont bien celles que vous pensez télécharger, et qu'elle sont aussi à"
echo "jour que vous pourriez l'espérer (testing et unstable ne devraient pas"
echo "être désynchronisées de plus d'un jour ou deux, stable-updates pas plus"
echo "de quelques semaines ou un mois)."
) | 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 "  * Pas de fichier Release au premier niveau"
              >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 Origine : $origline/$lablline"
              echo "  o Suite : $suitline/$codeline"
              echo "  o $dateline"
              echo "  o $dscrline"

              if [ "${dist%%/*}" != "$suitline" -a "${dist%%/*}" != "$codeline"
                      echo "  * Attention : $dist était demandée, $suitline/$cod
              fi

              lynx -reload -dump "${url}/dists/${dist}/Release.gpg" > /dev/null
              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 "  * Signé par ${err# } clef : ${rest#* }"
                          else
                              echo "  o Signé par : ${rest#* }"
                              okay=1
                          fi
                          err=""
                      elif [ "$gpgcode" = "BADSIG" ]; then
                          echo "  * Mauvaise signature par : ${rest#* }"
                          err=""
                      elif [ "$gpgcode" = "ERRSIG" ]; then
                          echo "  * Impossible de vérifier la signature par iden
                          err=""
                      elif [ "$gpgcode" = "SIGREVOKED" ]; then
                          err="$err Révoquée"
                      elif [ "$gpgcode" = "SIGEXPIRED" ]; then
                          err="$err Expirée"
```

```
                        fi
                    done
                    if [ "$okay" != 1 ]; then
                        echo "  * Pas de signature valable"
                        >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 "  * Problèmes avec $comp ($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 "  * Problèmes avec le composant $comp ($X, $
                        fi
                    fi
            done
            [ "$okaycomps" = "" ] || echo "  o Okay:$okaycomps"
            echo
      done

    echo "Résultat"
    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 "Les fichiers suivants de /var/lib/apt/lists n'ont pas été validés."
        echo "Cela peut soit être une simple indication inoffensive que ce script"
        echo "est bogué ou pas à jour, soit un indicateur de porte ouverte aux"
        echo "paquets de type chevaux de Troie sur le système."
        ) | fmt
        echo
        sed 's/^/    /' < UNVALIDATED
        echo
    fi

    if grep -q ^ BAD; then
```

```
    allokay=false
    (echo "Les contenus des fichiers suivants de /var/lib/apt/lists ne"
    echo "correspondent pas à ce qui était attendu. Cela peut signifier que"
    echo "ces sources ne sont pas à jour, qu'il y a un problème d'archive,"
    echo "ou que quelqu'un est en train d'utiliser le miroir pour distribuer"
    echo "des chevaux de Troie."
    if am_root; then
        echo "Les fichiers ont été renommés avec l'extension .FAILED et"
        echo "seront ignorés par 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 "Les fichiers suivants de /var/lib/apt/lists manquaient. Cela"
    echo "pourrait vous faire manquer des mises à jours de paquets vulnérables."
    ) | fmt
    echo
    sed 's/^/    /' < MISSING
    echo
fi

if grep -q ^ NOCHECK; then
    allokay=false
    (echo "Les contenus des fichiers suivants de /var/lib/apt/lists n'ont pas"
    echo "pu être validés à cause d'un manque de fichier Release signé, ou"
    echo "d'un manque d'entrée appropriée dans un fichier Release signé. Cela"
    echo "signifie probablement que les mainteneurs de ces sources sont"
    echo "négligents, mais pourrait signifier que ces sources sont en cours"
    echo "d'utilisation pour distribuer des chevaux de Troie."
    if am_root; then
        echo "Les fichiers ont été renommés avec l'extension .FAILED et"
        echo "seront ignorés par 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 'Tout semble se passer correctement !'
    echo
fi

rm -rf /tmp/apt-release-check
```

Vous pourriez devoir ajouter le correctif suivant pour *Sid* car **md5sum** ajoute un « - » après la somme quand l'entrée provient de l'entrée standard :

```
@@ -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"
```

Vérification de distribution pour les sources non Debian

Notez que, lors de l'utilisation de la dernière version d'apt (avec apt sécurisé), aucun effort supplémentaire ne devrait être nécessaire de votre part sauf si vous utilisez des sources non Debian, auquel cas une étape de confirmation supplémentaire sera imposée par apt-get. C'est évité en fournissant les fichiers `Release` et `Release.gpg` dans les sources non Debian. Le fichier `Release` peut être généré avec **apt-ftparchive** (disponible dans apt-utils 0.5.0 et ultérieur), le fichier `Release.gpg` est simplement une signature détachée. Pour générer les deux fichiers, suivez cette procédure simple :

```
$ rm -f dists/unstable/Release
$ apt-ftparchive release dists/unstable > dists/unstable/Release
$ gpg --sign -ba -o dists/unstable/Release.gpg dists/unstable/Release
```

Schéma alternatif de signature par paquet

Le schéma supplémentaire de signature de chacun des paquets permet aux paquets d'être vérifiés quand ils ne sont plus référencés par un fichier `Packages` existant, et également pour les paquets de tierce partie dont aucun `Packages` n'a jamais existé, pour qu'ils puissent être utilisés dans Debian, mais ce ne sera pas le schéma par défaut.

Ce schéma de signature des paquets peut être implémenté en utilisant debsig-verify et debsigs. Ces deux paquets peuvent signer et vérifier des signatures intégrées au .deb lui-même. Debian a déjà la capacité de faire cela actuellement, mais il n'y a aucun projet de mettre en place une charte ou d'autres outils puisque la signature de l'archive est préférée. Ces outils sont disponibles aux utilisateurs et aux administrateurs d'archive qui pourraient préférer utiliser ce schéma.

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.

Note : actuellement, `/etc/dpkg/dpkg.cfg` est livré avec « no-debsig » par défaut.

Seconde note : les signatures des développeurs sont actuellement enlevées lors de l'entrée du paquet dans l'archive car la méthode actuellement préférée est par vérification de distribution comme décrit précédemment.

Chapitre 8. Outils de sécurité dans Debian

FIXME : Besoin de plus de contenu.

Debian fournit un certain nombre d'outils qui peuvent rendre un système Debian apte à une utilisation sécurisée, y compris la protection des systèmes d'information au travers de pare-feu (qui agissent au niveau des paquets ou de la couche application), de systèmes de détection d'intrusions (basés sur le réseau ou sur l'hôte), d'évaluation des vulnérabilités, d'antivirus, de réseaux privés, etc.

Depuis Debian 3.0 (*Woody*), la distribution propose des logiciels de chiffrement intégrés à la distribution principale (*main*). OpenSSH et GNU Privacy Guard font partie de l'installation par défaut et le chiffrement fort est maintenant présent dans les navigateurs web, les serveurs web, les bases de données, etc. Une intégration plus poussée du chiffrement est prévue pour les versions ultérieures. Ces logiciels, à cause de restrictions d'exportation aux États-Unis, n'étaient pas distribués avec la distribution principale, mais inclus seulement dans les sites hors des États-Unis.

Outils d'évaluation des vulnérabilités à distance

Les outils fournis dans Debian pour effectuer une évaluation des vulnérabilités à distance sont : [1]

- nessus
- raccess
- nikto (en remplacement de **whisker**).

De loin l'outil le plus complet et mis à jour, nessus est composé d'un client (nessus) utilisé comme une interface graphique et d'un serveur (nessusd) qui lance les attaques programmées. Nessus connait des vulnérabilités à distance pour un grand nombre de systèmes y compris les appareils réseaux, les serveurs FTP, les serveurs HTTP, etc. Les dernières versions sont même capables de parcourir un site web et d'essayer de découvrir les pages interactives qui sont susceptibles d'être attaquées. Il existe également des clients Java et Win32 (non fournis dans Debian) qui peuvent être utilisés pour contacter le serveur de gestion.

nikto est un scanner pour évaluer les vulnérabilités d'un serveur HTTP et qui utilise des stratégies afin de contrer les systèmes de détection d'intrusions (IDS). Les IDS évoluant également, la plupart de ces techniques finissent par ne plus être efficace à titre d'*anti-IDS*). C'est tout de même l'un des meilleurs scanners disponibles pour tester les CGI et il est capable de détecter le serveur web utilisé afin de ne lancer les attaques que si elles ont des chances de fonctionner. De plus, la base de données utilisée pour scanner peut être facilement modifiée afin d'ajouter de nouveaux tests.

Outils pour parcourir le réseau

Debian fournit quelques outils pour parcourir des hôtes distants (toutefois en n'examinant pas les vulnérabilités). Ces outils sont, dans certains cas, utilisés comme des scanners de vulnérabilités. C'est le premier type d'« attaques » lancées contre des hôtes distants afin de tenter de déterminer les services disponibles. À l'heure actuelle, Debian fournit :

[1] Certains d'entre eux sont fournis en installant le paquet harden-remoteaudit.

- nmap
- xprobe
- p0f
- knocker
- isic
- hping2
- icmpush
- nbtscan (pour audits SMB ou NetBIOS)
- fragrouter
- **strobe** (dans le paquet netdiag) ;
- irpas

Même si xprobe ne permet que la détection des systèmes d'exploitation (en utilisant des empreintes TCP/IP), nmap et knocker font les deux : la détection du système d'exploitation et la détection de l'état des ports sur un système distant. D'un autre côté, hping3 et icmpush peuvent être utilisés dans le cadre d'attaques à distance par ICMP.

Conçu spécifiquement pour les réseaux SMB, nbtscan peut être utilisé pour scanner les réseaux IP et obtenir des informations sur les noms des serveurs ayant activé la prise en charge de NetBIOS, y compris l'adresse IP, le nom NetBIOS de l'ordinateur, les noms des utilisateurs connectés, les noms des réseaux, les adresses MAC, etc.

D'un autre côté, fragrouter peut être utilisé pour tester des systèmes de détection d'intrusion réseau et voir si le NIDS peut être éludé par des attaques par fragmentation (de paquets).

FIXME : Vérifier le http://bugs.debian.org/153117 (ITP fragrouter) pour voir s'il est inclus.

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).

Audits internes

De nos jours, seul l'outil tiger utilisé dans Debian peut être utilisé pour effectuer un audit interne (également appelé boîte blanche, « white box ») d'hôtes de façon à déterminer si le système de fichiers est installé correctement, les processus à l'écoute sur l'hôte, etc.

Contrôle du code source

Debian fournit plusieurs paquets qui peuvent être utilisés afin de contrôler le code source de programmes écrits en C ou C++ et d'identifier des erreurs de programmation qui pourraient conduire à des failles de sécurité exploitables :

- flawfinder

- rats
- splint
- pscan

Réseaux Privés Virtuels

Un réseau privé virtuel (VPN) est un groupe d'au moins deux ordinateurs, habituellement reliés à un réseau privé offrant un accès réseau public limité, qui communiquent de façon sécurisée par l'intermédiaire d'un réseau public. Les VPN peuvent connecter un seul ordinateur à un réseau privé (client serveur) ou un réseau local (LAN) distant à un réseau privé (serveur serveur). Les VPN incluent souvent l'utilisation du chiffrement, une authentification forte des utilisateurs ou hôtes distants et des méthodes pour cacher la topologie du réseau privé.

Debian fournit un nombre assez important de paquets pour mettre en place des réseaux privés virtuels chiffrés :

- vtun
- tunnelv (section non-US)
- cipe-source, cipe-common
- tinc
- secvpn
- pptpd
- openvpn
- openswan (http://www.openswan.org/).

FIXME : Mettre à jour cette information car elle a été écrite en pensant à FreeSWAN. Vérifier le bogue nº 237764 et le Message-Id: <200412101215.04040.rmayr@debian.org>.

Le paquet OpenSWAN est probablement le meilleur choix dans l'ensemble étant donné qu'il promet d'être fonctionnel avec tout matériel gérant le protocole de sécurité d'IP, IPsec (RFC 2411). Mais, les autres paquets peuvent vous aider à obtenir un tunnel sécurisé rapidement. Le protocole de tunnel point à point (PPTP) est le protocole propriétaire Microsoft pour les VPN. Il est pris en charge sous Linux mais il est connu pour avoir de sérieux problèmes de sécurité.

Pour plus d'informations, lire le http://www.traduc.org/docs/HOWTO/vf/VPN-Masquerade-HOWTO.html (couvre IPsec et PPTP), le http://www.traduc.org/docs/HOWTO/vf/VPN-HOWTO.html (couvre PPP à travers SSH), le http://www.linuxdoc.org/HOWTO/mini/Cipe+Masq.html et le http://www.linux-doc.org/HOWTO/mini/ppp-ssh/index.html.

Cela vaut également le coup de vérifier http://yavipin.sourceforge.net/, mais aucun paquet Debian ne semble être disponible pour l'instant.

Le tunnel point à point

Si vous désirez fournir un serveur de tunnel pour un environnement mixte (à la fois pour les systèmes d'exploitation Microsoft et les clients Linux) et qu'IPsec n'est pas une option (car il n'est fourni que pour

Windows 2000 et Windows XP), vous pouvez utiliser *PoPToP* (serveur de tunnel point à point), fourni dans le paquet pptpd.

Si vous voulez utiliser l'authentification et le chiffrage de Microsoft avec le serveur fourni dans le paquet ppp, veuillez noter la remarque suivante de la FAQ :

```
Utiliser PPP 2.3.8 n'est nécessaire que si vous voulez une
authentification et un chiffrement compatible Microsoft MSCHAPv2/MPPE.
La raison est que le correctif MSCHAPv2/MPPE actuellement fourni
(19990813) est relatif à PPP 2.3.8. Si vous n'avez pas besoin de
l'authentification ou du chiffrement compatible Microsoft, n'importe
quelle source PPP 2.3.x fera l'affaire.
```

Vous devez cependant appliquer le correctif noyau fourni par le paquet kernel-patch-mppe qui fournit le module pp_mppe pour pppd.

N'oubliez pas que le chiffrement dans ppptp vous oblige à stocker les mots de passe utilisateur en clair et que le protocole MS-CHAPv2 contient des http://mopo.informatik.uni-freiburg.de/pptp_mschapv2/.

Infrastructure de clefs publiques (PKI)

L'infrastructure de clefs publiques (PKI) est une architecture de sécurité introduite pour fournir un niveau de confiance amélioré lors de l'échange d'informations sur des réseaux non sécurisés. Elle utilise le concept de clefs de chiffrement publique et privée pour vérifier l'identité de l'expéditeur (signature) et garantir la confidentialité (chiffrement).

Lorsque vous vous intéressez aux PKI, vous vous trouvez confronté à une grande variété d'outils :

- une autorité de certification (Certificate Authority – CA) qui peut vous fournir des certificats extérieurs et travailler sous une hiérarchie donnée ;
- un répertoire pour conserver les certificats publics des utilisateurs ;
- une base de données pour maintenir une liste des certificats révoqués (Certificate Revocation Lists – CRL) ;
- des périphériques interopérants avec le CA pour éditer des cartes à puce, jetons USB ou n'importe quoi d'autre pour stocker les certificats en sécurité ;
- les applications prévues pour fonctionner avec des certificats de confiance peuvent utiliser des certificats distribués par des CA pour engager une communication chiffrée et vérifier les certificats délivrés contre un CRL (pour l'authentification et les solutions de signature complète unique) ;
- une autorité pour certifier les dates et signer numériquement des documents ;
- une console de gestion permettant une gestion correcte de tout cela (génération de certificats, contrôle de listes de révocation, etc.)

Debian GNU/Linux contient des paquets logiciels pour vous aider à résoudre ces problèmes de PKI, y compris **OpenSSL** (pour la génération de certificats), **OpenLDAP** (comme répertoire pour maintenir les certificats), **gnupg** et **openswan** (avec la prise en charge de la norme X.509). Cependant, le système d'exploitation ne fournit pas (depuis la version Woody, Debian 3.0) d'autorité de délivrance de certificat librement disponible comme pyCA, http://www.openca.org ou les exemples CA d'OpenSSL. Pour plus d'informations, reportez-vous au http://ospkibook.sourceforge.net.

Infrastructure SSL

Debian fournit quelques certificats SSL avec la distribution pour qu'ils puissent être installés localement. Ils sont disponibles dans le paquet ca-certificates. Ce paquet fournit un dépôt central des certificats qui ont été soumis à Debian et approuvé (c'est-à-dire vérifiés) par le responsable du paquet, cela est utile pour toutes les applications OpenSSL qui vérifient des connexion SSL.

FIXME : Lire debian-devel pour voir s'il y a quelque chose à ajouter à cela.

Outils antivirus

Il n'y a pas beaucoup d'antivirus fournis avec Debian, probablement parce que c'est un problème qui affecte très peu les utilisateurs de Linux. En fait, la plupart des antivirus disponibles sous Linux servent à protéger des ordinateurs fonctionnant sous un autre système d'exploitation. Cela s'explique par le modèle de sécurité UNIX qui fait une distinction entre les processus privilégiés (root) et les processus appartenant aux utilisateurs. Ainsi, un programme exécutable « hostile » qu'un utilisateur non privilégié a reçu ou créé et ensuite exécuté ne peut pas infecter ou d'une autre façon manipuler le système d'exploitation lui-même. Cependant, quelques virus et vers affectant Linux existent, même si aucun n'a jamais réussi à se répandre de façon significative sous Debian. Dans tous les cas, les administrateurs peuvent vouloir mettre en place des passerelles antivirus pour se protéger contre les virus affectant d'autres systèmes plus vulnérables dans leur réseau.

Debian GNU/Linux fournit à l'heure actuelle les outils suivants pour mettre en place des environnements antivirus.

- http://www.clamav.net, fourni depuis Debian *Sarge* (version 3.1). Des paquets sont fournis à la fois pour le scanneur de virus (clamav), pour le démon de scan (clamav-daemon) et pour les fichiers de données nécessaires au scanneur. Puisqu'un antivirus doit être à jour afin d'être vraiment utile, il y a trois moyens différents pour récupérer ces données : clamav-freshclam fournit un moyen de mettre à jour la base de données automatiquement par Internet, clamav-data fournit les fichiers de données directement.[2]

- mailscanner un scanneur de virus pour passerelle de courriels et un détecteur de pourriels. Fonctionnant avec sendmail, postfix ou exim, il peut utiliser plus de 17 types de scanneurs de virus différents dont clamav.

- libfile-scan-perl qui fournit File::Scan, une extension Perl pour scanner des fichiers à la recherche de virus. Ce module peut être utilisé pour créer un scanneur de virus indépendant de la plate-forme.

- http://www.sourceforge.net/projects/amavis, fourni par le paquet amavis-ng et disponible dans *Sarge*, est un scanneur de virus de courriel qui s'intègre avec différents serveurs de courriers (Exim, Sendmail, Postfix ou Qmail) et qui gère plus de 15 moteurs de recherche de virus (y compris clamav, File::Scan et openantivirus).

- http://packages.debian.org/sanitizer, un outil qui utilise le paquet procmail, qui peut filtrer les attachements de courrier, bloquer les attachements selon leurs noms de fichier et plus.

- http://packages.debian.org/amavis-postfix, un script qui fournit une interface depuis un MTA vers un ou plusieurs scanners commerciaux de virus (ce paquet est seulement construit pour le MTA **postfix**).

[2] Si vous utilisez ce dernier paquet et que vous utilisez une Debian officielle, la base de données ne sera pas mise à jour avec les mises à jour de sécurité. Vous devrez soit utiliser **clamav-getfiles** du paquet clamav-freshclam pour générer de nouveaux clamav-data ou mettre à jour depuis le dépôt des responsables :

```
deb http://people.debian.org/~zugschlus/clamav-data/ / deb-src http://people.debian.org/~zugschlus/clamav-data/
```

- exiscan, un scanneur de virus de courriel écrit en Perl qui fonctionne avec Exim.
- blackhole-qmail, un filtre de pourriel pour Qmail avec prise en charge intégrée pour Clamav.

Certains démons de passerelle proposent déjà des extensions d'outils pour construire des environnements antivirus, y compris exim4-daemon-heavy (la version *lourde* du MTA Exim), frox (un serveur mandataire FTP de cache transparent), messagewall (un démon mandataire SMTP) et pop3vscan (un mandataire POP3 transparent).

Présentement, clamav est l'unique scanneur d'antivirus inclus dans la branche officielle de Debian. En revanche, de nombreuses interfaces qui permettent d'utiliser l'antivirus avec des passerelles gérant différents protocoles sont offertes.

D'autres projets de logiciels libres d'antivirus qui pourraient être inclus dans une future version de Debian GNU/Linux : http://sourceforge.net/projects/openantivirus/ (consultez les bogues http://bugs.debian.org/150698 et http://bugs.debian.org/150695).

FIXME : Y a-t-il un paquet fournissant un script qui télécharge les dernières signatures de virus depuis http://www.openantivirus.org/latest.php ?

FIXME : Vérifier si scannerdaemon est le même que le démon scanner antivirus open (consultez les ITP).

Cependant, Debian ne fournira *jamais* de logiciels antivirus propriétaires et impossibles à redistribuer tels que : Panda Antivirus, NAI Netshield, http://www.sophos.com/, http://www.antivirus.com ou http://www.ravantivirus.com. Cela ne veut évidemment pas dire que ces logiciels ne peuvent pas être installés correctement sur un système Debian[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.

Agent GPG

Il est très courant de nos jours de signer numériquement (et parfois de chiffrer) des courriels. Vous pouvez, par exemple, trouver que de nombreuses personnes participant sur des listes de diffusion signent leur courriel de la liste. Les signatures numériques sont actuellement le seul moyen de vérifier qu'un message a été envoyé par l'expéditeur et non par une autre personne.

Debian GNU/Linux fournit un certain nombre de clients de messagerie avec des fonctionnalité de signature de courriels intégrés qui interagissent soit avec gnupg ou avec pgp :

- evolution.
- mutt.
- kmail.
- icedove (version sans marque de Mozilla Thunderbird) avec le greffon http://enigmail.mozdev.org/. Ce greffon est fourni par le paquet enigmail ;
- sylpheed. Selon la façon dont évolue la version stable de ce paquet, vous pouvez avoir besoin d'utiliser la *version dernier cri*, sylpheed-claws ;
- gnus, qui, lorsqu'il est installé avec le paquet mailcrypt, est une interface **emacs** à **gnupg** ;

[3] Un paquet nommé **f-prot-installer** est en fait un programme d'installation qui téléchargera le logiciel http://www.f-prot.com/products/home_use/linux/ pour l'installer sur le système. *F-prot* lui-même n'est pas libre, mais il est gratuit pour l'utilisation personnelle.

- kuvert, qui fournit cette fonctionnalité indépendamment du client de messagerie choisi en interagissant avec l'agent de transport de courrier (MTA).

Les serveurs de clefs permettent de télécharger des clefs publiques publiées pour pouvoir vérifier des signatures. Un tel serveur est http://wwwkeys.pgp.net. gnupg peut récupérer automatiquement des clefs publics qui ne sont pas déjà dans votre trousseau (keyring) public. Par exemple, pour configurer **gnupg** pour utiliser le serveur de clefs ci-dessus, modifiez le fichier `~/.gnupg/options` en ajoutant la ligne suivante : [4]

```
keyserver wwwkeys.pgp.net
```

La plupart des serveurs de clefs sont liés de tel sorte que, lorsqu'une clef publique est ajoutée à un serveur, l'addition soit propagée à tous les autres serveurs de clefs publiques. Le paquet debian-keyring fournit aussi les clefs publiques des développeurs Debian. Les trousseaux **gnupg** sont installés dans `/usr/share/keyrings/`.

Pour de plus amples renseignements :

- http://www.gnupg.org/documentation/faqs.fr.html
- http://www.gnupg.org/gph/fr/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.

[4] Pour plus d'exemples sur la façon de configurer **gnupg**, consultez `/usr/share/doc/mutt/examples/gpg.rc`.

Chapitre 9. Meilleures pratiques de sécurité pour les développeurs

Ce chapitre introduit certaines des meilleures pratiques de code sécurisé pour les développeurs écrivant des paquets Debian. Si vous êtes vraiment intéressé par le code sécurisé, vous devriez lire le http://www.dwheeler.com/secure-programs/ de David Wheeler et http://www.securecoding.org de Mark G. Graff et Kenneth R. van Wyk (O'Reilly, 2003).

Meilleures pratiques de vérification et conception sécurisées

Les développeurs qui empaquettent des logiciels devraient faire de leur mieux pour s'assurer que l'installation du logiciel, ou son utilisation, n'introduit pas de risques en matière de sécurité à la fois au système où il est installé et à ses utilisateurs.

Pour ce faire, ils devraient faire de leur mieux pour examiner le code source du paquet et détecter tous les défauts qui pourraient introduire des bogues de sécurité avant de publier le programme ou de distribuer une nouvelle version. Il est reconnu que le coût de correction de bogues augmente aux différentes étapes de son développement, il est donc plus facile (et moins coûteux) de corriger les bogues lors de la conception qu'une fois le logiciel déployé et en mode maintenance (plusieurs études disent que le coût dans cette dernière phase est *soixante* fois plus élevé). Bien que plusieurs outils essayent de détecter automatiquement ces défauts, les développeurs devraient faire leur possible pour se tenir au courant des différents types de défauts de sécurité afin de les comprendre et être capable de les remarquer dans le code qu'ils (ou d'autres) ont écrit.

Parmi les bogues de programmation qui conduisent à des bogues de sécurité, les plus typiques sont les http://fr.wikipedia.org/wiki/Dépassement_de_tampon, les dépassements de chaîne de formatage, les dépassements de tas et les dépassements d'entier (dans les programmes en C ou C++), les http://en.wikipedia.org/wiki/Symlink_race temporaires (dans les scripts), les http://en.wikipedia.org/wiki/Directory_traversal et les injections de commande (sur les serveurs) et http://fr.wikipedia.org/wiki/Cross-site_scripting, et les http://fr.wikipedia.org/wiki/Injection_SQL (dans le cas des applications orientées web). Pour de plus amples renseignements sur les bogues de sécurité, consultez la https://www.fortify.com/vulncat/en/vulncat/index.html de Fortify.

Certains de ces problèmes pourraient ne pas être faciles à repérer à moins d'être un expert dans le langage de programmation utilisé par le logiciel, mais certains problèmes sont faciles à détecter et à corriger. Par exemple, trouver des conditions de situation de compétitions temporaires à cause d'une mauvaise utilisation de répertoires temporaires peut se faire facilement en exécutant « `grep -r "/tmp/" .` ». Ces appels peuvent être examinés et les noms de fichiers écrits en dur, utilisant des répertoires temporaires, remplacés par des appels à **mktemp** ou **tempfile** dans les scripts d'interpréteur, File::Temp(3perl) dans les scripts Perl ou tmpfile(3) en C ou C++.

Certains outils permettent d'aider à l'examen de sécurité du code, comme rats, flawfinder et pscan. Pour de plus amples renseignements, consultez la http://www.debian.org/security/audit/tools.

Lors de l'empaquetage, les développeurs de logiciel doivent s'assurer de suivre les principes de sécurité habituels, y compris :

- le logiciel s'exécute avec le minimum de droits nécessaires :

  - le paquet installe des binaires setuid or setgid. **Lintian** avertira des binaires http://lintian.debian.org/reports/Tsetuid-binary.html, http://lintian.debian.org/reports/Tsetgid-binary.html ou http://lintian.debian.org/reports/Tsetuid-gid-binary.html ;

  - les démons fournis par le paquet s'exécutent avec un utilisateur aux droits restreints (consultez la section intitulée « Création d'utilisateurs et de groupes pour les démons logiciels ») ;

- les tâches programmées (c'est-à-dire **cron**) s'exécutant sur le système ne le sont *pas* en tant que superutilisateur, et si elle le sont, elles n'implémentent pas de tâches complexes.

Si vous devez faire l'un des deux, assurez-vous que les programmes qui pourraient s'exécuter avec des privilèges plus élevés ont été contrôlés pour les bogues de sécurité. En cas de doute, ou pour obtenir de l'aide, contactez l'http://www.debian.org/security/audit/. Pour les binaires setuid ou setgid, suivez la section de la charte Debian sur les http://www.debian.org/doc/debian-policy/ch-files.html#s-permissions-owners.

Pour de plus amples renseignements, spécifiques à la programmation sécurisée, assurez vous de lire (ou d'indiquer en amont) le http://www.dwheeler.com/secure-programs/ et le portail https://buildsecurityin.us-cert.gov/portal/.

Création d'utilisateurs et de groupes pour les démons logiciels

Si le logiciel exécute un démon qui n'a pas besoin des droits du superutilisateur, vous devez lui créer un utilisateur. Deux types d'utilisateurs Debian peuvent être utilisés par les paquets : avec identifiant (UID) statique (attribué par base-passwd ; consultez la section intitulée « Les utilisateurs et les groupes du système d'exploitation » pour une liste des utilisateurs statiques dans Debian) et avec identifiant dynamique dans l'intervalle dévolu aux utilisateurs système.

Dans le premier cas, il faut demander un identifiant de groupe ou d'utilisateur à base-passwd. Une fois l'utilisateur disponible, le paquet doit être distribué avec une dépendance sur la version adéquate du paquet base-passwd.

Dans le second cas, il faut créer un utilisateur système en *preinst* ou en *postinst* et rendre le paquet dépendant de `adduser (>= 3.11)`.

L'exemple de code suivant crée les utilisateur et groupe utilisés par le démon pour s'exécuter quand le paquet est installé ou mis à niveau :

```
[...]
case "$1" in
  install|upgrade)

  # Si le paquet a un fichier « default » permettant à
  # l'administrateur local d'écraser les valeurs par défaut

  [ -f "/etc/default/nompaquet" ] && . /etc/default/nompaquet

  # Valeurs par défaut correctes :

  [ -z "$SERVER_HOME" ] && SERVER_HOME=rép_serveur
  [ -z "$SERVER_USER" ] && SERVER_USER=utilisateur_serveur
  [ -z "$SERVER_NAME" ] && SERVER_NAME="Description du serveur"
```

```
  [ -z "$SERVER_GROUP" ] && SERVER_GROUP=groupe_serveur

  # Groupes auxquels l'utilisateur sera ajouté ; aucun si non défini.
  ADDGROUP=""

  # créer un utilisateur pour éviter d'exécuter le serveur en tant
  # que superutilisateur
  # 1. Créer le groupe s'il n'existe pas
  if ! getent group | grep -q "^$SERVER_GROUP:" ; then
     echo -n "Ajout du groupe $SERVER_GROUP.."
     addgroup --quiet --system $SERVER_GROUP 2>/dev/null ||true
     echo "fait"
  fi
  # 2. Créer un répertoire personnel s'il n'existe pas
  test -d $SERVER_HOME || mkdir $SERVER_HOME
  # 3. Créer un utilisateur s'il n'existe pas
  if ! getent passwd | grep -q "^$SERVER_USER:"; then
    echo -n "Ajout de l'utilisateur système $SERVER_USER.."
    adduser --quiet \
            --system \
            --ingroup $SERVER_GROUP \
            --no-create-home \
            --disabled-password \
            $SERVER_USER 2>/dev/null || true
    echo "fait"
  fi
  # 4. Ajuster l'entrée de mot de passe
  usermod -c "$SERVER_NAME" \
          -d $SERVER_HOME   \
          -g $SERVER_GROUP  \
             $SERVER_USER
  # 5. Ajuster les droits des fichiers et répertoires
  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. Ajouter l'utilisateur au groupe ADDGROUP
  if test -n $ADDGROUP
  then
      if ! groups $SERVER_USER | cut -d: -f2 | \
         grep -qw $ADDGROUP; then
           adduser $SERVER_USER $ADDGROUP
      fi
  fi
  ;;
  configure)

[...]
```

Assurez-vous que le fichier de script init.d :

- démarre le démon en abandonnant les droits du superutilisateur : si le logiciel ne fait pas l'appel setuid(2) ou seteuid(2) lui-même, l'option `--chuid` de **start-stop-daemon** est utilisable.

- n'arrête le démon que si l'identifiant utilisateur correspond, l'option `--user` de **start-stop-daemon** permet de faire cela.

- ne s'exécute pas si l'utilisateur ou le groupe n'existent pas :

```
if ! getent passwd | grep -q "^utilisateur_serveur:"; then
   echo "L'utilisateur du serveur n'existe pas. Abandon" >&2
   exit 1
fi
if ! getent group | grep -q "^groupe_serveur:" ; then
   echo "Le groupe du serveur n'existe pas. Abandon" >&2
   exit 1
fi
```

Si le paquet crée l'utilisateur système, il peut le retirer lors de la purge en *postrm*. Cela a cependant quelques inconvénients. Par exemple les fichiers créés par cet utilisateur seront orphelins et pourraient être repris par un nouvel utilisateur système plus tard si le même identifiant utilisateur lui est attribué[1]. Par conséquent, retirer les utilisateurs système lors de la purge n'est pas encore obligatoire et dépend des besoins du paquet. En cas de doute, cette action pourrait être faite en demandant à l'administrateur sa préfèrence lors du retrait du paquet (c'est-à-dire avec **debconf**).

Les responsables qui souhaitent supprimer des utilisateurs dans leurs scripts *postrm* sont renvoyés à l'option **deluser**/**deluser** `--system`.

L'exécution de programmes avec un utilisateur ayant des droits restreints assure qu'aucun problème de sécurité ne pourra endommager tout le système. Cela suit aussi le principe du *minimum de droits*. Songez aussi à limiter les droits dans les programmes avec d'autres mécanismes que l'exécution en tant que non superutilisateur[2]. Pour de plus amples renseignements, consultez le chapitre http://www.dwheeler.com/secure-programs/Secure-Programs-HOWTO/minimize-privileges.html du livre *HOWTO de programmation sécurisée pour Linux et UNIX*.

[1] Plusieurs discussions à propos de ces inconvénients ont déjà eu lieu comme http://lists.debian.org/debian-mentors/2004/10/msg00338.html et http://lists.debian.org/debian-devel/2004/05/msg01156.html.
[2] Vous pouvez même fournir une politique SELinux pour cela.

Chapitre 10. Avant la compromission

Maintenez le système sécurisé

Vous devriez faire tous les efforts nécessaires pour maintenir votre système sécurisé en surveillant son utilisation ainsi que les vulnérabilités qui pourraient l'affecter, en ajoutant les correctifs dès qu'ils sont disponibles. Même si vous avez installé un système vraiment sécurisé, vous devez garder à l'esprit que la sécurité d'un système se dégrade avec le temps. Des failles de sécurité peuvent être découvertes pour les services offerts et les utilisateurs peuvent affaiblir la sécurité du système soit à cause d'une incompréhension (par exemple, en accédant au système à distance à l'aide d'un protocole non chiffré, ou en utilisant des mots de passe faciles à deviner), ou parce qu'ils essaient activement de corrompre la sécurité du système (c'est-à-dire installer des services supplémentaires dans leur compte local).

Surveillance des failles de sécurité

Bien que la plupart des administrateurs ne soient conscients des failles de sécurité affectant leur système que lorsqu'un correctif est rendu disponible, vous pouvez être proactif et tenter de prévenir les attaques en introduisant des contre-mesures temporaires contre ces vulnérabilités dès que vous détectez qu'elles peuvent affecter le système. C'est particulièrement vrai sur un système exposé (c'est-à-dire connecté à Internet) et qui fournit un service. Dans ce cas, les administrateurs système devraient surveiller attentivement les sources d'informations connues pour être les premiers informés lorsqu'une faille pouvant affecter un service critique est détectée.

Cela signifie habituellement au moins s'abonner à la liste de diffusion des annonces, au site web du projet ou au système de suivi des bogues fourni par les développeurs pour les applications à surveiller. Par exemple, les utilisateurs d'Apache devraient surveiller régulièrement la http://httpd.apache.org/security_report.html et s'inscrire à la liste de diffusion des http://httpd.apache.org/lists.html#http-announce.

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.

Le système de suivi fournit des interfaces avec moteur de recherche (par nom http://cve.mitre.org et nom de paquet) et d'autres outils (comme debsecan, consultez la section intitulée « Vérification automatique des problèmes de sécurité avec debsecan ») utilisent ces bases de données pour fournir des informations sur les vulnérabilités qui n'ont pas encore été résolues pour un système donné.

Les administrateurs consciencieux peuvent utiliser ces renseignements pour déterminer les failles de sécurité pouvant affecter le système qu'ils gèrent, déterminer la sévérité du bogue et appliquer (si possible) des contre-mesures temporaires avant qu'un correctif soit disponible pour résoudre le problème.

Les problèmes de sécurité des versions suivies par l'équipe de sécurité de Debian devraient être traitées par une annonce de sécurité Debian (DSA) et seront disponibles pour tous les utilisateurs (consultez la section intitulée « Mettre à jour le système en permanence »). Une fois que les problèmes de sécurité sont résolus et annoncés, ils ne seront plus affichés par le système de suivi, mais vous pourrez encore chercher les vulnérabilités par leur nom CVE en utilisant la http://www.debian.org/security/crossreferences disponible pour les DSA publiées.

Remarquez cependant que les renseignements suivis par l'équipe de suivi en sécurité de testing ne concernent que les failles connues (c'est-à-dire déjà rendues publiques). Parfois, l'équipe de sécurité Debian peut gérer et préparer des DSA pour des paquets en fonction de renseignements non publics qu'ils ont obtenus sur des listes de diffusions restreintes, par le découvreur de la faille ou par les développeurs du logiciel. Ainsi, ne vous étonnez pas de découvrir des problèmes de sécurité dans une annonce qui ne sont jamais apparus dans le système de suivi des vulnérabilités.

Mettre à jour le système en permanence

Vous devriez effectuer des mises à jour de sécurité régulièrement. La plupart des stratagèmes sont basés sur des failles connues qui n'ont pas été corrigées à temps, comme l'explique ce http://www.cs.umd.edu/~waa/vulnerability.html (présenté lors du Symposium 2001 IEEE sur la sécurité et la confidentialité). Les mises à jour sont décrites dans la section intitulée « Faire une mise à jour de sécurité ».

Vérification par soi-même la disponibilité de mises à jour de sécurité

Debian dispose d'un outil spécifique pour déterminer si un système a besoin d'être mis à jour, mais beaucoup d'utilisateurs veulent simplement vérifier si des mises à jour de sécurité sont disponibles pour leur système.

Si vous avez configuré le système comme décrit en la section intitulée « Faire une mise à jour de sécurité », il suffit de faire :

```
# apt-get update
# apt-get upgrade -s
[ … passer en revue les paquets à mettre à jour… ]
# apt-get upgrade
# checkrestart
[ … redémarrer les services qui doivent être redémarrés… ]
```

Redémarrez ensuite les services dont les bibliothèques ont été mises à jour si c'est le cas. Remarque : consultez la section intitulée « Faire une mise à jour de sécurité » pour de plus amples renseignements sur les mises à jour de bibliothèques (et de noyau).

La première ligne téléchargera la liste des paquets disponibles depuis les sources de paquets configurées. L'option `-s` effectuera une simulation d'exécution, c'est-à-dire qu'elle ne va *pas* télécharger ou installer de paquets, mais qu'elle va plutôt signaler les paquets à télécharger ou installer. À partir de ce résultat, vous pouvez en déduire les paquets corrigés dans Debian et disponibles en mise à jour de sécurité. Par exemple :

```
# apt-get upgrade -s
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances... Fait
Calcul de la mise à jour... Fait
Les paquets suivants seront mis à jour :
  cvs libcupsys2
2 mis à jour, 0 nouvellement installés, 0 à enlever et 0 non mis à jour.
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).

Remarquez que le système doit être redémarré après une mise à jour du noyau.

Vérification de mises à jour sur station de travail

Depuis Debian 4.0 *Lenny*, Debian fournit et installe par défaut update-notifier. C'est une application GNOME qui est lancée lors de l'ouverture de la session et qui peut être utilisée pour faire le suivi des mises à jour disponibles pour le système et les installer. C'est fait en utilisant le paquet update-manager.

Pour un système stable, les mises à jour sont seulement disponibles quand un correctif de sécurité est disponible ou pour les versions intermédiaires. Par conséquent, si le système est configuré correctement pour recevoir les mises à jour de sécurité comme décrit en la section intitulée « Faire une mise à jour de sécurité » et qu'une tâche cron met à jour les informations sur les paquets, vous serez averti par une icône dans l'espace de notification du bureau.

La notification n'est pas intrusive et les utilisateurs ne sont pas forcés d'installer les mises à jour. Depuis l'icône de notification, un utilisateur du bureau (avec le mot de passe administrateur) peut accéder à une interface simple et voir les mises à jour disponibles puis de les installer.

Cette application fonctionne en consultant la base de données des paquets et en la comparant avec le système. Si cette base de données est mise à jour régulièrement par une tâche **cron**, alors son contenu sera plus récent que les paquets installés sur le système et l'application pourra vous avertir.

Apt installe une telle tâche cron (`/etc/cron.d/apt`) qui s'exécutera selon la configuration d'APT (plus spécifiquement *APT::Periodic*). Dans l'environnement GNOME, la valeur de la configuration peut être ajustée dans le menu Système > Administration > Sources de mise à jour > Mises à jour, ou en exécutant **/usr/bin/software-properties**.

Si le système télécharge quotidiennement la liste des paquets, mais ne télécharge pas les paquets eux-mêmes, le fichier `/etc/apt/apt.conf.d/10periodic` devrait ressembler à ceci :

```
APT::Periodic::Update-Package-Lists "1";
APT::Periodic::Download-Upgradeable-Packages "0";
```

Vous pouvez utiliser une tâche cron différente, comme celle installée par cron-apt (consultez la section intitulée « Vérification automatique des mises à jour avec cron-apt »). Vous pouvez aussi simplement vérifier vous-même les mises à jour en utilisant cette application.

Les utilisateurs de l'environnement KDE préféreront probablement installer adept et adept-notifier. Ils fournissent des fonctionnalités similaires, mais ne sont pas installés par défaut.

Vérification automatique des mises à jour avec cron-apt

Une autre méthode pour des mises à jour de sécurité automatiques est l'utilisation de cron-apt. Ce paquet fournit un outil pour mettre à jour le système à intervalles réguliers (en utilisant une tâche cron). Par défaut, il va simplement mettre à jour la liste des paquets et télécharger les nouveaux paquets. Il peut également être configuré pour envoyer un courrier à l'administrateur système.

Remarquez que vous pourriez vérifier la version de distribution comme décrit en la section intitulée « Vérification par version de distribution » pour mettre à jour automatiquement le système (même si vous ne té-

léchargez que les paquets). Sinon, vous ne pouvez pas être certain que les paquets téléchargés proviennent réellement d'une source de confiance.

Pour de plus amples renseignements, consultez le http://www.debian-administration.org/articles/162.

Vérification automatique des problèmes de sécurité avec debsecan

Le programme **debsecan** évalue l'état de la sécurité par rapport aux mises à jour de sécurité non effectuées et aux vulnérabilités sans correctif alors que cron-apt ne fournit qu'un rapport sur les mises à jour non effectuées. **debsecan** obtient les renseignements sur les failles qui ne sont pas corrigées à l'aide de la base de données des vulnérabilités qui est gérée par l'équipe de sécurité de Debian. Par conséquent, comme décrit en la section intitulée « Surveillance des failles de sécurité », il aide plus efficacement les administrateurs à suivre les failles de sécurité.

En installant le paquet debsecan, et si l'administrateur l'accepte, une tâche cron exécutera périodiquement **debsecan** et notifiera l'utilisateur choisi lorsqu'un paquet vulnérable est détecté. L'emplacement de la base de données des vulnérabilités est aussi paramétrable lors de l'installation et peut ensuite être modifié dans le fichier `/etc/default/debsecan`. C'est pratique pour les systèmes sans accès direct à Internet qui doivent télécharger les nouvelles informations depuis un miroir local pour avoir un seul chemin de mise à jour de la base de données des vulnérabilités.

Remarquez toutefois que l'équipe de sécurité suit beaucoup de failles, y compris des problèmes peu dangereux qui pourraient ne pas être corrigés lors des mises à jour de sécurité. De plus, certaines failles initialement considérées comme affectant Debian peuvent, plus tard et après enquête, être abandonnées. **debsecan** indiquera toutes les failles, ce qui peut en faire un outil plus verbeux que les autres outils décrits précédemment.

Pour plus d'informations, veuillez consulter le http://www.enyo.de/fw/software/debsecan/.

Autres méthodes de mises à jour de sécurité

Le paquet apticron, comme apt-cron, vérifiera les mises à jour et enverra des messages à l'administrateur. Pour plus d'informations, veuillez consulter le http://www.debian-administration.org/articles/491.

Vous pourriez également jeter un œil à http://clemens.endorphin.org/secpack/, un programme non officiel pour effectuer des mises à jour de sécurité depuis security.debian.org écrit par Fruhwirth Clemens, qui vérifie les signatures ou encore le module d'extension Nagios http://www.unixdaemon.net/nagios_plugins.html#check_debian_packages écrit par Dean Wilson.

Évitez la branche unstable

À moins de vouloir passer du temps à corriger les paquets vous-même quand une faille survient, vous ne devriez *pas* utiliser la branche unstable de Debian pour des systèmes en production. La raison principale est l'absence de mises à jour de sécurité pour *unstable*.

Certains problèmes de sécurité peuvent en fait apparaître dans unstable et *pas* dans la distribution *stable*. Cela est dû aux nouvelles fonctionnalités ajoutées constamment aux applications fournies, ainsi qu'aux nouvelles applications qui peuvent ne pas encore avoir été testées en profondeur.

Pour effectuer des mises à jour de sécurité dans la branche *unstable*, vous risquez de devoir faire des mises à jour complètes vers de nouvelles versions (ce qui peut mettre à jour beaucoup plus que les paquets touchés). Bien qu'il y ait des exceptions, les correctifs de sécurité sont habituellement rétroportés dans la branche *stable*. L'idée principale étant qu'entre les mises à jour, *aucun nouveau code* ne doit être ajouté, seulement des correctifs aux problèmes importants.

Remarquez que vous pouvez utiliser le système de suivi de sécurité (décrit en la section intitulée « Surveillance des failles de sécurité ») pour suivre les failles de sécurité affectant cette branche.

Suivi en sécurité de la branche testing

Si vous utilisez la branche *testing*, plusieurs problèmes sont à prendre en compte concernant la disponibilité des mises à jour de sécurité.

- Quand un correctif de sécurité est préparé, l'équipe de sécurité rétroporte le correctif pour *stable* (car stable est habituellement en retard de quelques versions mineures ou majeures). Le responsable du paquet s'occupe de préparer les paquets pour *unstable*, habituellement basé sur une nouvelle version amont. Parfois, les modifications se produisent en même temps et parfois l'une des distributions reçoit le correctif de sécurité avant. Les paquets de la distribution *stable* sont testés plus en profondeur que ceux d'*unstable* car ces derniers peuvent fournir la dernière version amont (qui pourrait ajouter de nouveaux bogues inconnus).
- Les mises à jour de sécurité sont disponibles pour la branche *unstable* quand le responsable du paquet crée une nouvelle version du paquet et pour *stable* quand l'équipe de sécurité effectue un envoi et publie une DSA. Veuillez noter que ni l'un, ni l'autre ne modifie *testing*.
- Si aucun (nouveau) bogue n'est détecté dans la version *unstable* de paquet, il est déplacé dans *testing* après plusieurs jours. Le délai est habituellement de dix jours, bien que cela dépende de la priorité de l'envoi des modifications et si l'entrée du paquet dans *testing* est bloquée par ses relations de dépendances. Notez que si l'entrée du paquet dans *testing* est bloquée, la priorité d'envoi ne changera pas le temps nécessaire pour y entrer.

Ce comportement peut changer selon l'état de publication de la distribution. Quand une nouvelle version est imminente, l'équipe de sécurité ou les responsables de paquet peuvent fournir des mises à jour directement dans *testing*.

De plus, l'http://secure-testing-master.debian.net peut publier des annonces de sécurité de testing (« Debian Testing Security Advisories » ou DTSA) pour les paquets de la branche *testing* si un problème de sécurité doit être immédiatement corrigé dans cette branche sans attendre la procédure normale (ou que la procédure normale est bloquée par d'autres paquets).

Les utilisateurs voulant tirer partie de ce suivi devraient ajouter les lignes suivante à `/etc/apt/sources.list` (au lieu des lignes indiqué en la section intitulée « Faire une mise à jour de sécurité ») :

```
    deb http://security.debian.org testing/updates main contrib non-free
# Cette ligne permet de télécharger aussi les paquets source
    deb-src  http://security.debian.org testing/updates main contrib non-free
```

Pour de plus amples renseignements sur ce suivi, veuillez lire l'http://lists.debian.org/debian-devel-announce/2006/05/msg00006.html. Ce suivi a officiellement commencé en http://lists.debian.org/debian-devel-announce/2005/09/msg00006.html dans un dépôt séparé avant d'être intégré à l'archive de sécurité principale.

Mises à jour automatiques dans un système Debian GNU/Linux

Tout d'abord, les mises à jour automatiques ne sont pas vraiment recommandées car les administrateurs devraient vérifier les DSA et comprendre l'impact de toute mise à jour de sécurité donnée.

Si vous voulez mettre à jour le système automatiquement, vous devriez suivre les conseils suivants.

- Configurer **apt** pour interdire la mise à jour des paquets à garder dans leur version actuelle, soit avec la fonctionnalité d'étiquetage (*pinning*) d'**apt**, soit en les marquant comme *hold* (à garder) avec **dpkg** ou **dselect**.

Pour conserver les paquets à une version donnée, vous devez éditer `/etc/apt/preferences` (consultez apt_preferences(5)) et ajouter :

```
Package: *
Pin: release a=stable
Pin-Priority: 100
```

FIXME : Vérifier si cette configuration est correcte.

- Utiliser soit cron-apt comme décrit dans la section intitulée « Vérification automatique des mises à jour avec cron-apt » et l'activer pour installer les paquets récupérés, soit ajouter une entrée **cron** vous-même pour exécuter la mise à jour quotidiennement, par exemple :

```
apt-get update && apt-get -y upgrade
```

L'option `-y` forcera **apt** à répondre automatiquement oui aux questions lors de la mise à jour. Dans certains cas, vous pourriez préférer l'option `--trivial-only` à `--assume-yes` (qui est équivalent de `-y`).[1]

- Configurer **cron** pour que **debconf** ne pose pas de question pendant les mises à jour, qui pourront ainsi être faites de façon non interactive.[2]

- Vérifier les résultats de l'exécution de **cron** envoyées au superutilisateur (sauf si la variable d'environnement `MAILTO` est modifiée dans le script).

Une alternative plus sûre peut être d'utiliser l'option `-d` (ou `--download-only`) pour télécharger les paquets nécessaires sans les installer. Puis, si l'exécution de **cron** indique que le système doit être mis à jour, cela peut être fait par l'administrateur.

Pour accomplir ces tâches, le système doit être configuré correctement pour télécharger les mises à jour de sécurité comme décrit en la section intitulée « Faire une mise à jour de sécurité ».

Cependant, cela n'est pas recommandé pour *unstable* sans analyse attentive, car vous pourriez placer le système dans un état inutilisable si un bogue sérieux s'introduit dans un paquet important et est installé sur le système. *testing* est un peu plus *sûre* de ce côté car les bogues sérieux ont une meilleure chance d'être détectés avant que le paquet n'entre dans la branche testing (cependant, vous pourriez n'avoir *aucune* mise à jour de sécurité disponible).

Si vous utilisez une distribution mixte, c'est-à-dire, une installation de *stable* avec des paquets mis à jour de *testing* ou d'*unstable*, vous pouvez jouer avec les préférences d'étiquetage et avec l'option `--target-release` d'**apt-get** pour ne mettre à jour *que* les paquets que de la nouvelle distribution.[3]

Tests d'intégrité périodiques

En vous basant sur les informations de base générées après l'installation (c'est-à-dire l'instantané décrit dans la section intitulée « Prendre un instantané («snapshot») du système »), vous pourriez effectuez un test

[1] Vous pourriez aussi utiliser l'option `--quiet` (`-q`) pour réduire la sortie d'**apt-get**, ce qui évitera la génération de message si aucun paquet n'est installé.
[2] Remarquez que certains paquets pourraient *ne pas* utiliser **debconf** et les mises à jour seront bloquées car les paquets attendront une réponse de l'administrateur pendant la configuration.
[3] C'est un problème courant car beaucoup d'utilisateurs veulent conserver un système stable tout en mettant à jour certains paquets avec *unstable* pour obtenir les dernières fonctionnalités. Ce besoin provient de l'évolution plus rapide de certains projets que le temps mis par Debian pour publier une nouvelle version *stable* de sa distribution.

d'intégrité de temps en temps. Un test d'intégrité pourra détecter des modifications du système de fichiers réalisées par un intrus ou dues à une erreur de l'administrateur système.

Les tests d'intégrité devraient, si possible, être réalisés non connectés.[4] C'est-à-dire, sans utiliser le système d'exploitation du système à contrôler, pour éviter un sentiment de sécurité erroné (c'est-à-dire des faux négatifs) produit, par exemple, par des rootkits installés. La base de données d'intégrité par rapport à laquelle le système est vérifiée devrait également être utilisée depuis un support en lecture seule.

Vous pouvez envisager de faire des vérifications d'intégrité en ligne en utilisant l'un des outils d'intégrité de système de fichiers disponibles (décrits dans la section intitulée « Vérifier l'intégrité des systèmes de fichiers ») s'il n'est pas possible de déconnecter le système. Cependant, des précautions devraient être prises pour utiliser une base de données d'intégrité en lecture seule et également pour assurer que les outils de vérification d'intégrité (et le noyau du système d'exploitation) n'ont pas été falsifiés.

Certains des outils mentionnés dans la section des outils d'intégrité, comme **aide**, **integrit** ou **samhain**, sont déjà préparés pour faire des vérifications périodiques (en utilisant la crontab dans les deux premiers cas et en utilisant un démon indépendant pour **samhain**) et ils peuvent avertir l'administrateur par différents moyens (habituellement par courriel, mais **samhain** peut également envoyer des pages, des alertes SNMP ou des alertes syslog) quand le système de fichiers est modifié.

Bien sûr, si vous exécutez une mise à jour de sécurité du système, l'instantané pris pour le système devrait être régénéré pour prendre en compte les modifications réalisées par la mise à jour de sécurité.

Mise en place de détection d'intrusion

Debian contient certains outils pour la détection d'intrusion qui permettent de défendre le système local ou d'autres systèmes du même réseau. Ce type de défense est important si le système est très critique ou si vous êtes vraiment paranoïaque. Les approches de détection d'intrusion les plus communes sont la détection statistique d'anomalies et la détection de correspondance de modèle.

Soyez toujours aux aguets de manière à réellement améliorer la sécurité du système avec n'importe lequel de ces outils, vous devez avoir un mécanisme d'alerte et réaction. Un système de détection d'intrusion est inutile si personne n'est prévenu.

Quand une attaque particulière est détectée, la plupart des outils de détection d'intrusion vont soit journaliser l'événement avec **syslogd**, soit envoyer des courriers au superutilisateur (le destinataire du courrier est habituellement configurable). Un administrateur doit configurer convenablement les outils pour éviter les fausses alertes. Les alertes peuvent également indiquer une attaque en cours et ne seraient pas très utiles un jour plus tard, puisque l'attaque pourrait déjà avoir été couronnée de succès. Assurez-vous donc qu'une règle de sécurité correcte a été mise en place vis-à-vis des alertes et que les mécanismes techniques pour l'implémenter sont en place.

Une source d'informations intéressante est la http://www.cert.org/tech_tips/intruder_detection_checklist.html.

Détection d'intrusion provenant du réseau

Les outils de détection d'intrusions provenant du réseau scrutent le trafic sur un segment de réseau et utilisent cette information comme source de données. Spécifiquement, les paquets du réseau sont examinés et ils sont vérifiés pour voir s'ils correspondent à une certaine signature.

[4] Une façon aisée de faire cela est d'utiliser un CD autonome (Live CD), comme http://www.knoppix-std.org/ contenant à la fois les outils d'intégrité de fichier et la base de donnée du système.

snort est un renifleur flexible de paquets ou un journaliseur qui détecte les attaques selon un dictionnaire de signatures d'attaque. Il détecte diverses attaques et sondes, comme des débordements de capacité, des scans dissimulés de ports, des attaques CGI, des sondes SMB, etc. **snort** dispose également d'une capacité d'alerte en temps réel. Vous pouvez utiliser **snort** pour un certain nombre d'hôtes du réseau ainsi que pour l'hôte local. Cet outil peut être installé sur n'importe quel routeur pour garder un œil sur le réseau. Installez-le simplement avec `apt-get install snort`, suivez les questions et surveillez ses journaux. Pour une infrastructure de sécurité un peu plus large, regardez http://www.prelude-ids.org.

Le paquet snort de Debian est installé avec de nombreuses vérifications de sécurité activées par défaut. Toutefois, vous devriez prendre le temps de personnaliser l'installation pour prendre en compte les services utilisés sur le système. Vous pourriez rechercher des vérifications supplémentaires spécifiques à ces services.

D'autres outils plus simples peuvent être utilisés pour détecter les attaques réseaux. portsentry est un paquet intéressant pour informer lorsqu'un scan du réseau est effectué sur site. D'autres outils comme ippl ou iplogger permettent de détecter certaines attaques IP (TCP et ICMP), même s'ils ne fournissent pas de techniques avancées pour détecter les attaques réseaux (comme le ferait **snort**).

Vous pouvez essayer chacun de ces outils avec le paquet Debian idswakeup, un générateur de fausses alertes et qui inclut un grand nombre de signature d'attaques communes.

Détection d'intrusion fondée sur l'hôte

La détection d'intrusion fondée sur l'hôte implique d'activer, sur le système à étudier, un logiciel qui utilise les journaux ou les programmes d'audit du système comme source de données. Il scrute les processus suspects, scrute les accès d'hôtes et peut même scruter les changements aux fichiers critiques du système.

tiger est un ancien outil de détection d'intrusion qui a été porté sous Debian depuis la distribution Woody. **tiger** fournit un ensemble de vérifications de problèmes communs liés aux failles de sécurité, il vérifie la robustesse des mots de passe, les problèmes de système de fichiers, les processus de communications et d'autres façons de compromettre le compte du superutilisateur. Ce paquet contient de nouvelles vérifications de sécurité spécifiques à Debian, y compris les vérifications de sommes de contrôle MD5 des fichiers installés, les emplacements de fichiers n'appartenant pas aux paquets et l'analyse des processus locaux à l'écoute. L'installation par défaut configure **tiger** pour être exécuté quotidiennement, en générant un compte-rendu envoyé au superutilisateur à propos des compromissions possibles du système.

Des outils d'analyse de journaux comme logcheck peuvent également être utilisés pour détecter des tentatives d'intrusions. Consultez la section intitulée « Utiliser et personnaliser **logcheck** ».

De plus, des paquets scrutant l'intégrité du système de fichiers (consultez la section intitulée « Vérifier l'intégrité des systèmes de fichiers ») peuvent être utiles dans la détection d'anomalies dans un environnement sécurisé. Une intrusion effective modifiera probablement certains fichiers du système de fichiers local pour court-circuiter les règles de sécurité locales, installer un cheval de Troie ou créer des utilisateurs. De tels événements peuvent être détectés avec les vérificateurs d'intégrité du système de fichiers.

Éviter les rootkits

Loadable Kernel Modules (LKM)

Les LKM (*Loadable Kernel Modules* ou modules de noyau chargeables) sont des fichiers contenant des composants de noyau chargeables dynamiquement utilisés pour étendre les fonctionnalités de noyau. Le principal avantage d'utiliser des modules est la possibilité d'ajouter des périphériques additionnels comme

une carte réseau ou une carte son sans avoir à recompiler le noyau entièrement. Cependant certains pirates peuvent utiliser les LKM pour les rootkits (knark et adore) afin d'installer des portes dérobées sur des systèmes GNU/Linux.

Les portes dérobées des LKM peuvent être plus sophistiquées et moins détectables que des rootkits traditionnels. Ils peuvent cacher des processus, des fichiers, des répertoires et même des connexions sans modifier les codes source des binaires. Par exemple, un LKM peut forcer le noyau à cacher des processus spécifiques dans `procps` pour que même une bonne copie du binaire **ps** ne puisse donner des informations exactes à propos des processus actuels du système.

Détection des rootkits

Il existe deux approches pour défendre le système contre les rootkits LKM, une défense proactive et une défense réactive. La détection peut être simple et sans douleur ou difficile et fatigante selon la mesure que vous choisissez.

Défense proactive

L'avantage de ce type de défense est qu'elle prévient des dommages que pourrait entraîner un rootkit au système. Une telle stratégie est de *les attraper en premier*, c'est-à-dire de charger un LKM bien défini pour protéger le système d'autres LKM infectés. Une deuxième stratégie consiste à retirer la fonctionnalité de chargement des modules du noyau lui-même. Notez, cependant, qu'il existe des rootkits qui peuvent fonctionner même dans ce cas, certains altèrent même directement `/dev/kmem` (la mémoire du noyau) pour se rendre indétectables.

Debian GNU/Linux fournit quelques paquets à utiliser pour mettre en place une défense proactive :

lcap — interface utilisateur agréable pour retirer les *fonctionnalités* (contrôle d'accès basé sur le noyau) dans le noyau, rendant le système plus sécurisé. Par exemple, exécuter `lcap CAP_SYS_MODULE` [5] enlèvera des fonctionnalités de chargement des modules (même pour le superutilisateur).[6] De vieilles informations sur ces fonctionnalités sont dans la section de Jon Corbet http://lwn.net/1999/1202/kernel.php3 sur LWN datant de décembre 1999.

Si vous n'avez pas besoin de toutes ces fonctionnalités de noyau sur un système GNU/Linux, vous pourriez désactiver la prise en charge des modules chargeables lors de la configuration du noyau. Pour désactiver la prise en charge des modules chargeables, positionnez simplement CONFIG_MODULES=n lors de l'étape de configuration de construction du noyau ou dans le fichier `.config`. Cela prévient des rootkits LKM mais vous ne pourrez plus utiliser les modules avec le noyau GNU/Linux. La désactivation des modules peut surcharger le noyau, rendant la gestion du chargement nécessaire.

Défense réactive

L'avantage d'une défense réactive est qu'elle représente une faible surcharge au niveau des ressources systèmes. Elle fonctionne en comparant la table des appels systèmes avec une copie sûre d'un fichier du disque, `System.map`. Bien sûr, une défense réactive n'avertira l'administrateur qu'après la compromission du système.

[5] 28 fonctionnalités existent, y compris : `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` et `CAP_SYS_TTY_CONFIG`. Elles peuvent être toutes désactivées pour renforcer le noyau.

[6] Vous n'avez pas besoin d'installer lcap pour faire cela, mais c'est plus facile que de configurer `/proc/sys/kernel/cap-bound` soi-même.

La détection des rootkits dans Debian peut être accomplie avec le paquet chkrootkit. Le programme http://www.chkrootkit.org cherche des signes de présence de plusieurs rootkits connus sur le système local, mais ce n'est pas un test définitif.

Idées géniales ou paranoïaques — ce que vous pourriez faire

C'est probablement la section la plus instable et la plus amusante, car j'espère que quelques unes des idées « bah, ça semble dingue » pourraient être réalisées. Vous trouverez ci-dessous certaines idées pour améliorer la sécurité — suivant votre point de vue vous les qualifierez de géniales, paranoïaques, folles ou inspirées.

- S'amuser avec PAM (Pluggable Authentication Modules). Conformément à l'article PAM du phrack 56, ce qui est bien avec PAM, c'est qu'« il n'est limité que par votre imagination ». C'est vrai. Imaginez une connexion de superutilisateur seulement possible avec empreinte digitale ou un scan de l'œil ou une cryptocarte (pourquoi ai-je fait une conjonction de OU et pas de ET ici ?).

- Journalisation fasciste. Je voudrais dire que tout ce dont nous avons discuté plus haut est de la « journalisation douce ». Si vous voulez effectuer une vraie journalisation, procurez-vous une imprimante avec du papier listing et journalisez tout en l'imprimant. Cela semble amusant, mais c'est fiable et ne peut être supprimé, ni altéré.

- Distribution CD. Cette idée est très simple à réaliser et offre une assez bonne sécurité. Créez une distribution Debian durcie, avec les règles de pare-feu adéquate, faites-en une image ISO amorçable et gravez-la sur un CD. Vous avez maintenant une bonne distribution en lecture seule avec environ 600 Mo d'espace pour les services. Assurez-vous juste que toutes les données qui devraient être écrites soient écrites sur le réseau. Il est impossible pour des intrus d'obtenir un accès en lecture et écriture sur ce système et toute modification réalisée par un intrus sera désactivée avec un redémarrage du système.

- Désactiver la prise en charge des modules. Comme décrit auparavant, une fois désactivée l'utilisation des modules du noyau à la compilation, beaucoup de portes dérobées basées sur le noyau sont impossibles à implémenter car la plupart d'entre elles sont basées sur l'installation de modules du noyau modifiés.

- Journalisation par câble série (contribution de Gaby Schilders). Tant que les serveurs ont des ports série, imaginez une machine dédiée à la journalisation pour un certain nombre de serveurs. Le système de journalisation serait déconnecté du réseau, et connecté aux serveurs par un multiplexeur de ports série (cyclades ou similaire). Maintenant faites journaliser vos serveurs par leurs ports série en écriture seule. La machine de journalisation n'accepterait que du texte en clair en entrée sur ses ports séries et n'écrirait que sur un fichier journal. Branchez un graveur de CD ou DVD et transférez-y les fichiers journaux quand le fichier journal atteint la capacité du support. Maintenant il ne manque plus qu'un graveur avec chargeur de CD automatique… Pas autant « copie en dur » que la journalisation directe vers l'imprimante, mais cette méthode peut gérer de larges volumes et les CD prennent moins d'espace de stockage.

- Modifiez les attributs de tous les fichiers avec **chattr** (tiré du Tips-HOWTO écrit par Jim Dennis). Tout de suite après avoir installé et configuré initialement le système, utilisez le programme **chattr** avec l'attribut `+i` pour rendre les fichiers non-modifiables (le fichier ne peut être supprimé, renommé, lié ou réécrit). Envisagez de positionner cet attribut sur tous les fichiers de `/bin`, `/sbin/`, `/usr/bin`, `/usr/sbin`, `/usr/lib` et tous les fichiers noyau de la racine. Vous pouvez également faire une copie de tous les fichiers de `/etc/`, en utilisant **tar**, et marquer l'archive comme immuable.

  Cette stratégie permettra de limiter les dégâts possibles une fois connecté en superutilisateur. Cela empêchera d'écraser des fichiers avec un opérateur de redirection mal placé, de rendre le système inutili-

sable avec une espace mal placée dans une commande **rm -rf** (il est toujours possible de faire pas mal de dégâts aux données, mais les bibliothèques et binaires seront mieux protégés).

Cela limite aussi la réalisation d'un grand nombre d'exploitations de faille de sécurité et de dénis de service (car beaucoup d'entre eux dépendent de l'écrasement d'un fichier par les actions d'un programme SETUID qui *ne fournit aucune invite de commandes*).

Le seul inconvénient de cette stratégie survient lorsque vous compilez et installez divers binaires systèmes. D'un autre côté, cela empêche aussi le **make install** d'écraser les fichiers. Quand vous oubliez de lire le Makefile et de faire un **chattr -i**, les fichiers qui vont être réécrits (et les répertoires auxquels vous voulez ajouter des fichiers) # la commande make échoue, utilisez juste la commande **chattr** et relancez-le. Vous pouvez aussi profiter de l'occasion pour déplacer vos vieux binaires et bibliothèques dans un répertoire .old/ ou dans une archive tar par exemple.

Remarquez que cette stratégie empêche aussi de mettre à jour les paquets du système car les fichiers existants ne peuvent être remplacés, vous pourriez donc avoir un mécanisme pour désactiver l'attribut immuable sur tous les binaires juste avant de faire un **apt-get update**.

- Couper 2 ou 4 fils du câble réseau afin de rendre les communications UDP unidirectionnelles. Ensuite, utilisez des paquets UDP pour envoyer des informations à la machine destinatrice qui peut agir en tant que serveur de journalisation sécurisé ou système de stockage de carte de crédit.

Construction d'un pot de miel

Un pot de miel est un système conçu pour apprendre aux administrateurs système les techniques de sondage et d'exploitation des attaquants. Il s'agit d'une configuration système qui a pour but d'être sondée, attaquée et potentiellement exploitée. En apprenant les outils et méthodes utilisées par l'attaquant, un administrateur système peut apprendre à mieux protéger ses propres systèmes et son réseau.

Un système Debian GNU/Linux peut facilement être configuré comme un pot de miel, si vous y consacrez le temps de l'implémenter et de le surveiller. Vous pouvez facilement mettre en place le serveur de pot de miel factice ainsi que le pare-feu[7] qui contrôle le pot de miel et un certain type de détecteur d'intrusion réseau, placez-le sur Internet et attendez. Prenez soin de vous assurer d'être averti à temps (consultez la section intitulée « L'importance des journaux et des alertes ») si le système est victime d'une exploitation pour que vous puissiez prendre des mesures appropriées et mettre un terme à la compromission après en avoir assez vu. Voici quelques paquets et problèmes à considérer lors de la configuration de pot de miel :

- la technologie pare-feu dont vous aurez besoin (fournie par les noyaux Linux) ;
- syslog-ng pour envoyer les journaux du pot de miel vers un serveur de journalisation système distant ;
- snort pour configurer la capture de tout le trafic réseau arrivant sur le pot de miel et détecter les attaques ;
- osh, un interpréteur de commande restreint à sécurité améliorée et SETUID root avec journalisation (consultez l'article de Lance Spitzner référencé ci-dessous) ;
- tous les démons à utiliser pour le serveur factice pot de miel. Selon le type d'attaque que vous voulez analyser, vous renforcerez *ou non* le pot de miel et vous le conserverez ou non à jour avec les mises à jour de sécurité ;
- des vérificateurs d'intégrité (consultez la section intitulée « Vérifier l'intégrité des systèmes de fichiers ») et la boîte à outils du légiste (The Coroner's Toolkit (tct)) pour faire des audits après l'attaque ;

[7] Vous utiliserez généralement un pare-feu pont pour que le pare-feu lui-même ne soit pas détectable, consultez la section intitulée « Configuration d'un pare-feu pont ».

- honeyd et farpd pour mettre en place un pot de miel qui écoutera les connexions vers des adresses IP non utilisées et les fera suivre vers des scripts simulant des services actifs. Regardez également iisemulator ;

- tinyhoneypot pour mettre en place un serveur pot de miel simple avec des services factices.

Si vous ne pouvez pas utiliser des systèmes de réserve pour construire les pots de miel et les systèmes réseau pour le protéger et le contrôler, vous pouvez utiliser la technologie de virtualisation disponible dans **xen** ou **uml** (User-Mode-Linux). Si vous choisissez cette route, vous devrez modifier le noyau soit avec kernel-patch-xen, soit avec kernel-patch-uml, ou encore installer l'un des noyaux précompilés disponibles depuis Debian Lenny.

Vous pouvez en lire plus sur la construction des pots de miel dans l'excellent article http://www.net-security.org/text/articles/spitzner/honeypot.shtml de Lance Spizner (dans la série des « connaissez votre ennemi »). De même, le http://project.honeynet.org/ fournit des informations utiles sur la façon de configurer un pot de miel et de contrôler les résultats d'une attaque.

Chapitre 11. Après la compromission (la réponse à l'incident)

Comportement général

Si vous êtes physiquement présent quand l'attaque se déroule et que faire ce qui suit n'a pas d'effet fâcheux sur vos transactions commerciales, la première réaction devrait être de débrancher simplement la machine du réseau en débranchant la carte réseau. La désactivation du réseau à la première couche est le seul vrai moyen de garder un attaquant en dehors d'une machine compromise (conseil avisé de Phillip Hofmeister).

Cependant, certains outils installés à l'aide d'un *rootkit*, d'un cheval de Troie ou même d'un utilisateur malhonnête connecté par une porte dérobée (backdoor), pourraient être capables de détecter cet évènement et d'y réagir. Voir un `rm -rf /` s'exécuter au moment de débrancher le réseau du système n'est pas vraiment très drôle. Si vous ne désirez pas prendre ce risque et que vous êtes certain que le système est compromis, vous devriez *débrancher le câble d'alimentation* (voire tous, s'il y en a plusieurs) et croiser les doigts. Cela peut sembler extrême, mais en fait cela désamorcera toute bombe à retardement que l'intrus pourrait avoir programmé. Dans ce cas, le système compromis *ne doit pas être redémarré*. Soit le disque dur devrait être déplacé sur un autre système pour analyse, soit vous devriez utiliser un autre support (un CD) pour amorcer le système et pour l'analyser. Vous *ne devriez pas* utiliser les disquettes de récupération de Debian pour amorcer le système, mais vous *pouvez* utiliser l'invite de commande fournie par les disquettes d'installation (Alt+F2 pour l'atteindre) pour analyser[1] le système.

La méthode recommandée pour récupérer un système compromis est d'utiliser un CD autonome avec tous les outils (et les modules du noyau) dont vous pouvez avoir besoin pour accéder au système compromis. Vous pouvez utiliser le paquet mkinitrd-cd pour construire un tel CD[2]. Vous pourriez également trouver le CD http://www.caine-live.net/ (Computer Aided Investigative Environment) utile ici, car il s'agit d'un CD autonome activement développé avec des outils d'analyse post mortem utiles dans ces situations. Il n'y a pas (encore) d'outil basé sur Debian comme celui-ci, ni de moyen simple de construire un CD en utilisant sa propre sélection de paquets Debian et mkinitrd-cd (vous devrez donc lire la documentation fournie avec celui-ci pour faire vos propres CD).

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/.

Certaines questions générales sur la façon de gérer un système Debian GNU/Linux compromis sont également disponibles dans la section intitulée « Le système est vulnérable ! (En êtes-vous certain ?) ».

Copies de sauvegarde du système

Rappelez-vous que si vous êtes certain que le système a été compromis, vous ne pouvez pas faire confiance aux logiciels qui s'y trouvent ou à n'importe quelle autre information qu'il vous donne. Les applications pourraient dissimuler un cheval de Troie, des modules pourraient être installés dans le noyau, etc.

[1] Si vous êtes aventureux, vous pouvez vous connecter au système et sauver les informations sur tous les processus en fonctionnement (vous en aurez beaucoup dans /proc/nnn/). Il est possible d'avoir l'intégralité du code exécutable depuis la mémoire, même si l'attaquant a supprimé les fichiers exécutables du disque. Puis tirez sur le cordon d'alimentation.

[2] En fait, c'est l'outil utilisé pour construire les CD pour le projet http://www.gibraltar.at/ (un pare-feu sur un CD autonome basé sur la distribution Debian).

La meilleure chose à faire est une sauvegarde complète du système de fichiers (en utilisant **dd**) après avoir démarré depuis un support sûr. Les CD Debian GNU/Linux peuvent être utiles pour cela, car une console en mode texte est disponible dans le deuxième terminal une fois l'installateur démarré (allez-y en pressant CTRL+ALT+F2 suivi de la touche « Entrée »). À partir de cette console, sauvegardez les informations ailleurs si possible (éventuellement sur un serveur de fichiers par NFS ou FTP). Par la suite, vous pourrez analyser les informations pendant que le système compromis est hors-ligne ou réinstallé.

Si vous êtes certain que la seule compromission est un cheval de Troie dans l'un des modules du noyau, vous pouvez tenter d'exécuter le noyau à partir du CD en mode *rescue*. Assurez-vous aussi de démarrer en mode *single user* de façon à ce qu'aucun autre cheval de Troie ne s'exécute après le redémarrage.

Contacter le CERT local

Le CERT (*Computer and Emergency Response Team*) est une organisation qui peut vous aider à récupérer un système compromis. Il y a des CERT partout dans le monde [3] et vous devriez contacter le CERT local en cas d'incident de sécurité qui a conduit à une compromission système. Les personnes du CERT local peuvent vous aider à le récupérer.

Fournir au CERT (ou au centre de coordination CERT) des informations sur la compromission, même si vous ne demandez pas d'aide, peut également aider d'autres personnes car les informations agrégées des incidents reportés sont utilisées pour déterminer si une faille donnée est répandue, s'il y a un nouveau ver dans la nature, les nouveaux outils d'attaque utilisés. Ces renseignements sont utilisés pour fournir à la communauté Internet des informations sur les http://www.cert.org/current/ et pour publier des http://www.cert.org/incident_notes/ et même des http://www.cert.org/advisories/. Pour des informations plus détaillées sur la façon (et les raisons) de rendre compte d'un incident, veuillez consulter les http://www.cert.org/tech_tips/incident_reporting.html.

Vous pouvez également utiliser un mécanisme moins formel si vous avez besoin d'aide pour récupérer un système compromis ou si vous voulez discuter d'informations d'incident, comme la http://marc.theaimsgroup.com/?l=incidents et la http://marc.theaimsgroup.com/?l=intrusions.

Analyse post mortem

Si vous souhaitez rassembler plus d'informations, le paquet tct (The Coroner's Toolkit de Dan Farmer et Wietse Venema) contient des utilitaires pour effectuer une analyse post mortem d'un système. tct permet à l'utilisateur de collecter des informations sur les fichiers effacés, les processus qui s'exécutent et plus. Consultez la documentation fournie pour plus d'informations. Ces utilitaires, ainsi que quelques autres, sont disponibles dans http://www.sleuthkit.org/ de Brian Carrier. Ils permettent l'analyse post mortem d'une image des disques par une interface Web. Dans Debian, vous trouverez les paquets sleuthkit (les outils) et autopsy (l'interface graphique).

N'oubliez pas que l'analyse post mortem devrait toujours être faite sur une copie des données et *jamais* sur les données elles-mêmes. Si ces dernières sont altérées par cette analyse, vous pourriez perdre des indices importants pour comprendre ce qui s'est passé exactement, en plus de rendre les preuves potentiellement non recevables en cour.

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.porcu-

[3] Voici une liste de quelques CERT. Pour la liste complète, consultez le http://www.first.org/about/organization/teams/index.html (FIRST est le *Forum of Incident Response and Security Teams*) : http://www.auscert.org.au (Australie), http://www.unam-cert.unam.mx/ (Mexique) http://www.cert.funet.fi (Finlande), http://www.dfn-cert.de (Allemagne), http://cert.uni-stuttgart.de/ (Allemagne), http://security.dico.unimi.it/ (Italie), http://www.jpcert.or.jp/ (Japon), http://cert.uninett.no (Norvège), http://www.cert.hr (Croatie) http://www.cert.pl (Pologne), http://www.cert.ru (Russie), http://www.arnes.si/si-cert/ (Slovénie) http://www.rediris.es/cert/ (Espagne), http://www.switch.ch/cert/ (Suisse), http://www.cert.org.tw (Taïwan) et http://www.cert.org (États-Unis).

pine.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 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 : Ce paragraphe fournira, dans un avenir proche je l'espère, plus d'informations sur l'analyse post mortem d'un système Debian.

FIXME : Décrire comment utiliser debsums sur un système stable avec les md5sums sur un CD et le système de fichiers récupéré restauré sur une partition séparée.

FIXME : Ajouter des liens vers des articles d'analyse post mortem (tel que le défi inversé de Honeynet ou les http://staff.washington.edu/dittrich/).

Analyse des programmes malveillants (malware)

D'autres outils permettant l'analyse post mortem sont disponibles pour la distribution Debian : strace et ltrace.

L'un de ces paquets peut être utilisé pour analyser des binaires dangereux (comme des portes dérobées) afin de déterminer comment ils fonctionnent et ce qu'ils font au système. **ldd** (dans libc6), **strings** et **objdump** (tous deux dans binutils) font aussi partie des outils fréquemment utilisés.

Pour faire l'autopsie de binaires suspects ou contenant des portes dérobées récupérés d'un système compromis, vous devriez utiliser un environnement sécurisé (par exemple, dans une image bochs, xen ou un environnement **chroot**é en utilisant un compte ayant peu de droits[4]). Le système pourrait être victime de la porte dérobée et compromis à son tour si vous ne prenez pas garde !

Si vous êtes intéressé par les programmes malveillants, alors vous devriez lire le chapitre http://www.porcupine.org/forensics/forensic-discovery/chapter6.html du livre http://www.porcupine.org/forensics/forensic-discovery/ de Dan Farmer et Wietse Venema.

[4] Faites *très* attention si vous utilisez **chroot**, car si le programme utilise une faille de sécurité au niveau du noyau afin d'accroître ses droits, il pourrait tout de même réussir à compromettre le système.

Chapitre 12. Foire Aux Questions (FAQ)

Ce chapitre introduit quelques questions qui reviennent souvent sur la liste de diffusion de sécurité. Vous devriez les consulter avant de poster sur la liste ou certains pourraient vous dire d'aller RTFM.

La sécurité dans le système d'exploitation Debian

Debian est-elle plus sûre que X ?

Un système est aussi sûr que l'administrateur est capable de le rendre. Debian essaie d'installer les services d'une façon *sûre par défaut*, mais elle n'est peut-être pas aussi paranoïaque que d'autres systèmes d'exploitation qui installent tous les services *désactivés par défaut*. Toutefois, l'administrateur système a besoin d'adapter la sécurité du système à la politique de sécurité locale.

Pour une liste des données concernant les failles de sécurité pour plusieurs systèmes d'exploitation, consultez les http://www.cert.org/stats/cert_stats.html ou générez des statistiques en utilisant la http://nvd.nist.gov/statistics.cfm (anciennement ICAT). Est-ce que ces données sont utiles ? Plusieurs facteurs sont à considérer pour l'interprétation des données, mais remarquez qu'elles ne permettent pas de comparer les failles d'un système d'exploitation par rapport à un autre.[1] Rappelez-vous également que certaines failles signalées concernant Debian ne s'appliquent qu'à la branche *unstable* (c'est-à-dire non publiée).

Debian est-elle mieux sécurisée que d'autres distributions Linux (comme Red Hat, SuSE, etc.) ?

Peu de grandes différences existent entre les distributions Linux, à l'exception de l'installation de base et du système de gestion des paquets. La plupart des distributions partagent une grande partie des mêmes applications, les différences étant seulement dans les versions de ces applications livrées avec la version stable de la distribution. Par exemple, le noyau, BIND, Apache, OpenSSH, Xorg, gcc, zlib, etc. sont tous communs entre les distributions Linux.

Par exemple, Red Hat a joué de malchance et a livré une version stable quand truc était en version 1.2.3, où une faille sécurité a été découverte plus tard. Debian, d'un autre côté, a été plus chanceuse de livrer truc 1.2.4, qui contient la correction du bogue. Cela a été le cas avec le gros problème de http://www.cert.org/advisories/CA-2000-17.html il y quelques années.

Beaucoup de collaboration existe entre les équipes de sécurité respectives des distributions Linux majeures. Les mises à jour de sécurité connues sont rarement, voire jamais, laissées non corrigées par un distributeur. La connaissance d'une faille de sécurité n'est jamais cachée à un autre distributeur, tout comme les corrections sont habituellement coordonnées en amont ou par le http://www.cert.org. Par conséquent, les mises à jour de sécurité nécessaires sont habituellement diffusés en même temps et la sécurité relative des différentes distributions est très semblable.

L'un des principaux avantages de Debian concernant la sécurité est la facilité des mises à jour du système par l'utilisation d'**apt**. Voici quelques autres aspects de la sécurité dans Debian à considérer.

[1] Par exemple, à partir de certaines données, Windows NT semblerait plus sûr que Linux, ce qui est une assertion discutable. Après tout, les distributions Linux fournissent habituellement beaucoup plus d'applications par rapport à Windows NT de Microsoft. Ces problèmes de *failles comptabilisées* sont mieux décrits dans http://www.dwheeler.com/oss_fs_why.html#security de David A. Wheeler.

- Debian fournit plus d'outils de sécurité que les autres distributions, consultez Chapitre 8, *Outils de sécurité dans Debian*.
- L'installation standard de Debian est plus petite (moins de fonctionnalités) et donc plus sûre. Les autres distributions, au nom de l'utilisabilité, ont tendance à installer plusieurs services par défaut et parfois, ils ne sont pas configurés correctement (rappelez-vous de http://www.sophos.com/virusinfo/analyses/linuxlion.html et http://www.sophos.com/virusinfo/analyses/linuxramen.html). L'installation de Debian n'est pas aussi limitée que celle d'OpenBSD (aucun démon n'est activé par défaut), mais c'est un bon compromis.[2]
- Debian documente les meilleures pratiques de sécurité dans des documents comme celui-ci.

De nombreux bogues Debian sont dans Bugtraq, cela la rend-elle plus vulnérable ?

Debian distribue un grand nombre, en augmentation constante, de paquets logiciels, probablement plus que la plupart des systèmes d'exploitation propriétaires. Par conséquent le risque est plus grand de trouver des logiciels victimes de failles de sécurité exploitables que sur les systèmes contenant moins de logiciels.

De plus en plus de personnes examinent le code source à la recherche de failles. De nombreux annonces sont liées à des audits de code source effectués sur des composants logiciels majeurs livrés dans Debian. Lorsqu'un de ces audits de code source fait ressortir une faille majeur, elle est réparée et une alerte est envoyée aux listes comme celle de BugTraq.

Les bogues présents dans la distribution Debian affectent également d'autres distributeurs et distributions. Vérifiez la partie « Debian specific: yes/no » en haut de chaque annonce (DSA).

Debian possède-t-elle une certification relative à la sécurité ?

Réponse courte : non.

Réponse longue : la certification coûte de l'argent (particulièrement, une certification de sécurité *sérieuse* et personne n'a attribué de ressources pour faire certifier la distribution Debian GNU/Linux à n'importe quel niveau que ce soit, par exemple, la Common Criteria. Si vous êtes intéressé par l'obtention d'une distribution GNU/Linux certifiée, essayez de fournir les ressources pour que cela devienne possible.

Au moins deux distributions Linux sont actuellement certifiées à différents niveaux http://fr.wikipedia.org/wiki/Evaluation_Assurance_Level. Remarquez que certains des tests CC sont en cours d'intégration dans le http://ltp.sourceforge.net disponible dans le paquet Debian ltp.

Existe-t-il un programme de durcissement pour Debian ?

Oui. http://bastille-linux.sourceforge.net/, orienté à la base vers certaines distributions Linux (Red Hat et Mandrake), cela fonctionne actuellement aussi sur Debian. Des étapes sont prévues pour intégrer les changements de la version amont dans le paquet Debian nommé bastille.

Certains pensent, cependant, qu'un outil de durcissement n'élimine pas la nécessité d'une bonne administration.

Je veux fournir le service XYZ, lequel dois-je choisir ?

L'une des grandes forces de Debian est la grande variété de choix disponibles entre les paquets fournissant la même fonctionnalité (serveurs DNS, serveurs de messagerie, serveurs FTP, serveurs web, etc.). Cela

[2] Sans diminuer le fait que d'autres distributions, comme Red Hat ou Mandrake, prennent aussi en compte la sécurité dans leurs installations standard en demandant à l'utilisateur de sélectionner des *profils de sécurité* ou en utilisant des assistants pour configurer des *pare-feu personnels*.

peut être déroutant pour l'administrateur débutant lorsqu'il essaie de déterminer l'outil adapté à son besoin. Le meilleur choix dans une situation donnée dépend d'un équilibre entre les fonctionnalités et la sécurité nécessaires. Voici quelques questions à se poser pour choisir parmi des paquets semblables.

- Est-ce que le logiciel est maintenu en amont ? De quand date la dernière version ?
- Le paquet est-il mûr ? Le numéro de version n'indiquera vraiment *rien* concernant sa maturité. Essayez de tracer l'histoire du logiciel.
- Est-ce que le logiciel est truffé de bogues ? Y a-t-il eu des alertes de sécurité liées au logiciel ?
- Est-ce que le logiciel fournit toutes les fonctionnalités nécessaires ? Fournit-il plus que le nécessaire ?

Comment mieux sécuriser le service XYZ dans Debian ?

Les informations disponibles dans ce document vous permettront de rendre certains services (FTP, BIND) plus sécurisés dans Debian GNU/Linux. Toutefois, pour les services non abordés ici, vous pouvez vérifier la documentation des programmes ou les informations générales sur Linux. La plupart des directives concernant la sécurité des systèmes UNIX peut également s'appliquer à Debian. Ainsi, la sécurisation d'un service X dans Debian revient, la plupart du temps, à sécuriser un service dans n'importe quelle autre distribution Linux (ou UNIX, peu importe).

Comment supprimer toutes les informations de version pour les services ?

Si vous ne voulez pas que des utilisateurs se connectent au démon POP3, par exemple, et récupèrent des renseignements sur le système, vous pourriez supprimer (ou modifier) les versions affichées aux utilisateurs. [3] Faire cela dépend du logiciel que vous utilisez pour un service donné. Par exemple, dans **postfix**, vous pouvez placer la bannière SMTP suivante ans `/etc/postfix/main.cf` :

```
smtpd_banner = $myhostname ESMTP $mail_name (Debian/GNU)
```

D'autres logiciels ne sont pas aussi faciles à modifier. SSH devra être recompilé pour pouvoir modifier la version affichée. Prenez garde à ne pas supprimer la première partie (`SSH-2.0`) de la bannière, car les clients l'utilisent pour identifier les protocoles pris en charge par le paquet.

Les paquets Debian sont-ils tous sûrs ?

L'équipe de sécurité Debian ne peut pas analyser tous les paquets inclus dans Debian pour tester des potentielles failles de sécurité, simplement à cause du manque de ressources pour contrôler le code source de l'ensemble du projet. Cependant Debian bénéficie des audits de code source réalisés par des développeurs amont.

De fait, un développeur Debian pourrait distribuer un cheval de Troie dans un paquet sans moyen de le vérifier. Même s'il était introduit dans une branche Debian, il serait impossible de couvrir toutes les situations imaginables dans lesquelles le cheval de Troie pourrait agir. C'est pourquoi Debian a une clause de licence de *« non garantie »*.

Cependant, les utilisateurs Debian peuvent être assurés que le code stable rassemble une large audience et que la plupart des problèmes seront découverts pendant l'utilisation. Installer des logiciels non testés dans un système critique n'est pas recommandé (si vous ne pouvez pas fournir l'audit de code nécessaire). Dans tous les cas, si des failles de sécurité étaient intégrées à la distribution, le processus permettant d'inclure

[3] Notez que c'est de la « sécurité par l'obscurité » et ne vaudra probablement pas l'effort à long terme.

les paquets (utilisation de signature numérique) assure que le problème pourra être remonté jusqu'au développeur, et que le projet Debian ne prend pas cela à la légère.

Pourquoi certains fichiers journaux ou fichiers de configuration sont-ils lisibles par tous les utilisateurs, est-ce que c'est sûr ?

Vous pouvez bien sûr modifier les permissions Debian par défaut du système. La règle actuelle concernant les fichiers journaux et les fichiers de configuration est qu'ils doivent être lisibles par tous les utilisateurs *sauf* s'ils fournissent des informations sensibles.

Soyez attentifs si vous faites des modifications car :

- des processus pourraient ne plus pouvoir écrire dans des fichiers journaux si leurs permissions ont été restreintes ;
- certains applications peuvent ne pas fonctionner si le fichier de configuration dont elles dépendent est illisible. Par exemple, si vous supprimez la permission en lecture pour tous les utilisateurs de `/etc/samba/smb.conf`, le programme **smbclient** ne pourra pas fonctionner pour un utilisateur normal.

FIXME : Vérifier si c'est écrit dans la Charte. Certains paquets (par exemple, les démons FTP) semblent nécessiter différentes permissions.

Pourquoi est-ce que /root/ (ou UserX) a 755 comme permissions ?

De fait, la même question s'applique pour tout autre utilisateur. Comme l'installation de Debian ne place *aucun* fichier dans ce répertoire, il n'y a aucune information sensible à y protéger. Si vous pensez que ces permissions sont trop laxistes pour le système, vous pouvez les renforcer en 750. Pour les utilisateurs, veuillez lire la section intitulée « Limiter l'accès aux informations d'autres utilisateurs ».

This Debian security mailing list http://lists.debian.org/debian-devel/2000/11/msg00783.html has more on this issue.

Après l'installation de grsec ou d'un pare-feu, j'ai commencé à recevoir beaucoup de messages de console ! Comment les supprimer ?

Si vous recevez des messages en console et que `/etc/syslog.conf` est configuré pour les rediriger dans des fichiers ou dans un TTY spécial, vous pourriez voir des messages envoyés directement en console.

Le niveau de journalisation en console par défaut est 7 quel que soit le noyau, donc tous les messages avec une priorité inférieure apparaîtront dans la console. En général, les pare-feu (la règle LOG) et d'autres outils de sécurité journalisent à des priorités inférieures donc les messages sont envoyés directement en console.

Pour réduire les messages envoyés en console, vous pouvez utiliser **dmesg** (l'option `-n`, consultez dmseg(8)>), qui examine et *contrôle* le tampon anneau du noyau. Pour corriger cela après le prochain redémarrage, modifiez `/etc/init.d/klogd` en substituant :

```
KLOGD=""
```

par :

```
KLOGD="-c 4"
```

Utilisez un nombre plus petit pour `-c` si vous les voyez toujours. Une description des différents niveaux de journalisation est disponible dans `/usr/include/sys/syslog.h` :

```
#define LOG_EMERG   0  /* le système est inutilisable */
#define LOG_ALERT   1  /* une action doit être entreprise immédiatement */
#define LOG_CRIT    2  /* conditions critiques */
#define LOG_ERR     3  /* conditions d'erreur */
#define LOG_WARNING 4  /* conditions d'avertissement */
#define LOG_NOTICE  5  /* normal, mais conditions significatives */
#define LOG_INFO    6  /* informatif */
#define LOG_DEBUG   7  /* messages de débogage */
```

Les utilisateurs et les groupes du système d'exploitation

Tous les utilisateurs systèmes sont-ils nécessaires ?

Oui et non. Debian est livrée avec certains utilisateurs prédéfinis (identifiant utilisateur (UID) < 99 comme décrit dans la http://www.debian.org/doc/debian-policy/ ou `/usr/share/doc/base-passwd/README`) afin de faciliter l'installation de certains services qui imposent d'être lancés par un utilisateur ayant un UID approprié. Si vous n'avez pas l'intention d'installer de nouveaux services, vous pouvez supprimer sans problème ces utilisateurs qui ne possèdent aucun fichier sur le système et n'exécutent aucun service. Dans tous les cas, le comportement par défaut est que les UID de 0 à 99 sont réservées dans Debian et les UID de 100 à 999 sont crées par des paquets lors de l'installation (et supprimées quand le paquet est purgé).

Vous pouvez facilement trouver les utilisateurs ne possédant aucun fichier en exécutant la commande suivante[4] (assurez-vous de l'exécuter en tant que superutilisateur, étant donné qu'un utilisateur ordinaire pourrait ne pas avoir les droits nécessaires pour accéder à certains répertoires sensibles) :

```
cut -f 1 -d : /etc/passwd |
while read i; do find / -user "$i" | grep -q . || echo "$i"; done
```

Ces utilisateurs sont fournis par base-passwd. Vous trouverez dans sa documentation plus d'informations sur la manière dont ces utilisateurs sont gérés dans Debian. Voici la liste des utilisateurs par défaut (avec un groupe correspondant).

- root : c'est (typiquement) le superutilisateur.
- daemon : quelques démons sans droit ont besoin de pouvoir écrire certains fichiers du disque en tant que daemon:daemon (par exemple, **portmap**, **atd**, et probablement d'autres). Les démons qui n'ont besoin d'aucune appartenance de fichier peuvent tourner en tant que nobody:nogroup, et des démons plus complexes ou plus consciencieux de la sécurité tournent en tant qu'utilisateurs spécifiques. L'utilisateur daemon est aussi utile pour les démons installés localement.
- bin : maintenu pour des raisons historiques.
- sys : comme bin. Toutefois, /dev/vcs* et `/var/spool/cups` appartiennent au groupe sys.
- sync : l'interpréteur de commandes de l'utilisateur sync est /bin/sync. Donc, si son mot de passe est quelque chose de facile à deviner (comme « »), n'importe qui peut synchroniser le système depuis la console même sans compte sur le système.
- games : de nombreux jeux sont SETGID à games pour pouvoir écrire dans les fichiers des meilleurs scores. C'est expliqué dans la Charte.

[4] Prenez garde, car cela parcourt tout le système. Si vous avez beaucoup de disques et partitions, vous pourriez réduire sa portée.

- man : le programme man est (parfois) lancé en tant qu'utilisateur man, il peut alors écrire les pages cat vers `/var/cache/man`.
- lp : utilisé par les démons d'impression.
- mail : les boîtes aux lettres de `/var/mail` appartiennent au groupe mail, comme décrit dans la Charte. L'utilisateur et le groupe sont également utilisés à d'autres fins par différents MTA.
- news : plusieurs serveurs de nouvelles et autres programmes associés (comme **suck**) utilisent l'utilisateur et le groupe news de différentes façons. Les fichiers dans la file d'attente des nouvelles appartiennent souvent à l'utilisateur et au groupe news. Les programmes comme **inews** qui peuvent être utilisés pour envoyer des nouvelles sont typiquement SETGID news.
- uucp : l'utilisateur et le groupe uucp sont utilisés par le sous-système UUCP. Les fichiers de file d'attente et de configuration lui appartiennent. Les utilisateurs du groupe uucp peuvent exécuter **uucico**.
- proxy : comme daemon, cet utilisateur et ce groupe sont utilisés par certains démons (en particulier les démons de mandataire) qui ne possèdent pas d'identifiant utilisateur et qui n'ont pas besoin de posséder des fichiers. Par exemple, le groupe proxy est utilisé par **pdnsd** et **squid** est exécuté en tant qu'utilisateur proxy.
- majordom : **majordomo** a un identifiant utilisateur alloué statiquement sur les systèmes Debian pour des raisons historiques. Il n'est plus installé sur les nouveaux systèmes.
- postgres : les bases de données **postgresql** appartiennent à cet utilisateur et ce groupe. Tous les fichiers dans `/var/lib/postgresql` appartiennent à cet utilisateur afin d'imposer un niveau de sécurité convenable.
- www-data : certains serveurs web tournent en tant que www-data. Le contenu web *ne devrait pas* appartenir à cet utilisateur, sinon un serveur Internet compromis serait en mesure de réécrire un site web. Les données transférées par les serveurs web, incluant les fichiers journaux, seront la propriété de www-data.
- backup : de cette manière la responsabilité de sauvegarde ou restauration peut être localement déléguée à quelqu'un sans avoir à lui donner tous les droits du superutilisateur.
- operator : c'est historiquement (et pratiquement) le seul compte « utilisateur » qui peut se connecter à distance, sans dépendre de NIS ou NFS.
- list : les archives des listes de diffusion et les données appartiennent à cet utilisateur et à son groupe. Certains programmes de listes de diffusion utilisent aussi cet utilisateur.
- irc : utilisé par les démons IRC. Un utilisateur alloué statiquement est nécessaire à cause d'un bogue dans **ircd**, il se SETUID lui-même vers un UID donné au démarrage.
- gnats.
- nobody, nogroup : les démons qui n'ont pas besoin d'être propriétaires de fichiers devraient fonctionner sous l'utilisateur nobody et le groupe nogroup. Donc, aucun fichier sur un système ne devrait appartenir à cet utilisateur ou à ce groupe.

Les autres groupes suivants n'ont pas d'utilisateur associé.

- adm : utilisé pour les tâches de surveillance du système. Les membres de ce groupe peuvent lire de nombreux journaux d'événements dans `/var/log` et peuvent utiliser xconsole. Historiquement, `/var/log` était `/usr/adm` (et plus tard `/var/adm`) d'où le nom du groupe.
- tty : les périphériques TTY appartiennent à ce groupe. C'est utilisé par **write** et `wall` pour leur permettre d'écrire sur les TTY d'autres personnes.

- disk : accès brut aux disques. Quasiment équivalent à l'accès superutilisateur.
- kmem : `/dev/kmem` et les fichiers similaires sont lisibles par ce groupe. C'est la plupart du temps un reste de BSD, mais certains programmes en ont besoin pour un accès direct en lecture sur la mémoire du système ce qui peut ainsi être fait par SETGID kmem.
- dialout : accès direct et total aux ports séries. Les membres de ce groupe peuvent reconfigurer les modems, téléphoner n'importe où, etc.
- dip : le nom du groupe signifie « Dialup IP ». Être dans le groupe dip permet d'utiliser des outils comme **ppp**, **dip**, **wvdial**, etc. pour établir une connexion. Les utilisateurs de ce groupe ne peuvent pas configurer le modem, ils peuvent juste utiliser les programmes qui en font usage.
- fax : autorise les membres à utiliser les logiciels de fax pour envoyer et recevoir des faxes.
- voice : boîte vocale, utile pour les systèmes qui utilisent les modems comme répondeurs.
- cdrom : utilisé localement pour donner à certains utilisateurs un accès aux lecteurs de CD.
- floppy : utilisé localement pour donner à certains utilisateurs un accès aux lecteurs de disquettes.
- tape : utilisé localement pour donner à certains utilisateurs un accès aux lecteurs de bandes.
- sudo : les membres de ce groupe n'ont pas besoin de fournir un mot de passe lors de l'utilisation de **sudo**. Consultez `/usr/share/doc/sudo/OPTIONS`.
- audio : utilisé localement pour donner à certains utilisateurs un accès aux périphériques audio.
- src : ce groupe possède les codes source, y compris les fichiers de `/usr/src`. Il peut être utilisé pour permettre à un utilisateur de manipuler les codes source du système.
- shadow : `/etc/shadow` est lisible par ce groupe. Certains programmes ayant besoin d'accéder à ce fichier sont SETGID shadow.
- utmp : les membres de ce groupe peuvent écrire dans `/var/run/utmp` et dans fichiers similaires. Les programmes qui nécessitent l'écriture sont SETGID utmp.
- video : utilisé localement pour donner à certains utilisateurs un accès aux périphériques vidéo.
- staff : autorise les utilisateurs à ajouter des modifications au système local (`/usr/local`, `/home`) sans avoir les droits du superutilisateur. À comparer au groupe « adm » plus apparenté à la surveillance et la sécurité.
- users : alors que les systèmes Debian utilisent le système de groupe privé par utilisateur par défaut (chaque utilisateur a son propre groupe), certains préfèrent d'utiliser un système de groupes plus traditionnel. Dans ce système, chaque utilisateur est un membre de ce groupe.

J'ai supprimé un utilisateur système ! Comment puis-je le récupérer ?

Si vous avez supprimé un utilisateur système et que vous n'avez pas de sauvegardes des fichiers `password` et `group`, vous pouvez essayez de récupérer de ce problème en utilisant **update-passwd** (consultez update-passwd(8)).

Quelle est la différence entre les groupes adm et staff ?

Le groupe « adm » est normalement celui des administrateurs et leur permet de lire les journaux d'activités sans utiliser **su**. Le groupe « staff » est généralement pour les administrateurs système secondaires afin de faire des choses dans `/usr/local` et de créer des répertoires dans `home`.

Pourquoi y a-t-il un nouveau groupe à chaque ajout de nouvel utilisateur (ou pourquoi Debian attribue-t-elle un groupe à chaque utilisateur) ?

Le comportement par défaut dans Debian est que chaque utilisateur a son propre groupe privé. Le schéma traditionnel UNIX place tous les utilisateurs dans le groupe *users*. Des groupes supplémentaires étaient créés et utilisés pour restreindre l'accès à des fichiers partagés associés aux différents répertoires de projets. La gestion des fichiers devenait difficile quand un seul utilisateur travaillait sur plusieurs projets car quand quelqu'un créait un fichier, ce dernier était associé au groupe primaire auquel il appartenait (c'est-à-dire « users »).

Le schéma Debian résout ce problème en attribuant à chaque utilisateur son propre groupe ; ainsi avec un umask correct (0002) et le bit SETGID positionné dans un répertoire de projet donné, le groupe correct est automatiquement attribué aux fichiers créés dans ce répertoire. Cela facilite le travail sur plusieurs projets sans modifier les groupes ou les umask pour travailler sur des fichiers partagés.

Vous pouvez, cependant, changer ce comportement en modifiant `/etc/adduser.conf`. Changez la variable *USERGROUPS* à « no », pour qu'aucun nouveau groupe ne soit créé quand un nouvel utilisateur est créé. Positionnez également *USERS_GID* au GID du groupe users auquel appartiennent tous les utilisateurs.

Questions concernant les services et les ports ouverts

Pourquoi tous les services sont-ils activés lors de l'installation ?

C'est un compromis entre la présence de sécurité et la facilité d'utilisation. Contrairement à OpenBSD, qui désactive tous les services non activés par l'administrateur, Debian GNU/Linux activera tous les services installés à moins de les désactiver (consultez la section intitulée « Désactivation de services démon » pour plus de renseignements). Après tout, vous avez installé ces services de votre propre chef, n'est-ce pas ?

De nombreuses discussions sur les listes de diffusion Debian (sur debian-devel et debian-security) ont eu lieu sur l'installation standard. Cependant, il n'y a pas de consensus à ce jour (mars 2002) sur la solution à adopter.

Puis-je retirer inetd ?

inetd n'est pas aisé à retirer étant donné que netbase dépend du paquet qui le fournit (netkit-inetd). Si vous voulez le retirer, vous pouvez soit le désactiver (consultez la section intitulée « Désactivation de services démon »), soit retirer le paquet en utilisant equivs.

Pourquoi le port 111 est-il ouvert ?

Le port 111 est le mappeur de port sunrpc, il est installé par défaut dans toutes les installations de base d'un système Debian puisqu'il est nécessaire pour savoir quand le programme d'un utilisateur a besoin de RPC pour fonctionner correctement. Dans tous les cas, il est principalement utilisé pour NFS. Si vous n'en avez pas besoin, retirez-le comme décrit en la section intitulée « Sécurisation des services RPC ».

Dans les versions du paquet portmap ultérieures à 5-5, le portmapper peut être en fait installé en n'écoutant que localhost (en modifiant `/etc/default/portmap`).

À quoi sert identd (port 113) ?

Le service identd est un service d'authentification du propriétaire d'une connexion TCP/IP spécifique au serveur distant acceptant la connexion. Par exemple, quand un utilisateur se connecte sur un hôte distant, **inetd** de l'hôte distant va envoyer une demande sur le port 113 pour déterminer les informations du pro-

priétaire. C'est souvent utilisé pour les serveurs de courriers, FTP et IRC et peut également être utilisé pour remonter la trace de l'utilisateur qui attaque un système distant par l'intermédiaire de votre machine.

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).

Des services utilisent les ports 1 et 6, quels sont ces services et comment les enlever ?

Si la commande `netstat -an` affiche :

```
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
-
```

Aucun processus n'écoute sur les ports 1 et 6. En fait, un processus écoute sur une socket *raw* (brut) pour les protocoles 1 (ICMP) et 6 (TCP). Un tel comportement est courant pour les chevaux de Troie et pour certains systèmes de détection d'intrusions comme iplogger et portsentry. Si ces paquets sont installés, supprimez-les simplement. Sinon, essayez l'option `-p` (processus) de netstat pour voir le processus à l'écoute.

Le port XYZ est ouvert, puis-je le fermer ?

Bien sûr que vous pouvez, les ports laissés ouverts doivent adhérer à la politique de sécurité du site concernant les services publiques disponibles pour les autres systèmes. Vérifiez s'ils sont ouvert par **inetd** (consultez la section intitulée « Désactivation d'inetd ou de ses services ») ou par d'autres paquets installés et prenez les mesures adéquates (par exemple, configuration d'inetd, suppression du paquet, éviter qu'il démarre au démarrage).

Est-ce que la suppression de services de /etc/services va aider à sécuriser la machine ?

Non, le fichier `/etc/services` fournit juste une cartographie d'un nom virtuel à un numéro de port donné. La suppression des noms ne va pas (en général) empêcher les services d'être lancées. Certains démons ne se lanceront peut-être pas si `/etc/services` est modifié mais ce n'est pas la norme. Pour désactiver correctement les services, consultez la section intitulée « Désactivation de services démon ».

Problèmes courants de sécurité

J'ai perdu mon mot de passe et je ne peux plus accéder au système !

Les démarches pour récupérer le système dépendent des différentes procédures appliquées pour limiter l'accès à **lilo** et au BIOS.

Si les deux accès sont limités, vous devez désactiver les fonctionnalités du BIOS (démarrer uniquement depuis le disque dur) avant de commencer. Si vous avez également oublié le mot de passe du BIOS, vous devrez ouvrir le système et retirer manuellement la pile du BIOS.

Une fois activé l'amorçage depuis un CD ou une disquette, vous pouvez essayer de :

- démarrer depuis une disquette de secours (rescue) et démarrer le noyau ;
- accéder aux consoles virtuelles (Alt+F2) ;
- monter le disque dur où est placé la partition /root ;
- éditer (la disquette de secours de Debian 2.2 est livrée avec **ae**, Debian 3.0 est livrée avec **nano-tiny** qui est similaire à **vi**) `/etc/shadow` et modifier la ligne :

  ```
  root:asdfjl290341274075:XXXX:X:XXXX:X::: (X=n'importe quel nombre)
  ```

  par :

  ```
  root::XXXX:X:XXXX:X:::
  ```

Cela retirera le mot de passe superutilisateur oublié, contenu dans le premier champ séparé par deux points après le nom d'utilisateur. Enregistrez le fichier, redémarrer le système et connectez-vous en tant que superutilisateur (avec un mot de passe vide). Cela fonctionnera sauf si le système est configuré plus strictement, par exemple sans autorisation des connexions avec mot de passe vide ou des connexions du superutilisateur à partir de la console.

Si ces caractéristiques ont été introduites, vous devrez passer en mode utilisateur unique. Si LILO a été restreint, vous devrez relancer **lilo** après la réinitialisation du superutlisateur précédente. C'est assez rusé puisque `/etc/lilo.conf` devra être modifié car le système de fichiers racine est alors un disque virtuel et non le vrai disque dur.

Une fois que LILO n'est plus restreint, vous pouvez :

- presser l'une des touches Alt, Maj ou Ctrl juste avant que le BIOS système ne finisse, pour obtenir l'invite de LILO ;
- entrer `linux single`, `linux init=/bin/sh` ou `linux 1` à l'invite ;
- cela donnera accès à une invite de commandes un mode utilisateur unique (un mot de passe sera demandé, mais vous le connaissez déjà) ;
- remonter en lecture/écriture la partition racine (/), en utilisant la commande de montage :

  ```
  mount -o remount,rw /
  ```

- modifier le mot de passe du superutilisateur avec **passwd** (étant superutilisateur, l'ancien mot de passe ne sera pas demandé).

Comment mettre en place un service pour les utilisateurs sans leur donner un compte avec invite de commande ?

Par exemple, si vous voulez mettre en place un service POP, vous n'avez pas besoin de configurer un compte d'utilisateur pour chaque utilisateur y accédant. Il est préférable de mettre en place une authentification basé sur un répertoire grâce à un service externe (comme Radius, LDAP ou une base de données SQL). Installez simplement la bibliothèque PAM appropriée (libpam-radius-auth, libpam-ldap, libpam-pgsql ou libpam-mysql), consultez la documentation (pour commencer, consultez la section intitulée

« Authentification utilisateur: PAM ») et configurez le service en activant PAM pour utiliser la méthode que vous avez choisi. C'est fait en éditant les fichiers de `/etc/pam.d/` pour les services et en modifiant :

```
auth   required    pam_unix_auth.so shadow nullok use_first_pass
```

en, par exemple pour ldap :

```
auth   required    pam_ldap.so
```

Dans le cas de répertoires LDAP, certains services fournissent des schémas LDAP à inclure dans le répertoire et qui sont nécessaires pour utiliser l'authentification LDAP. Si vous utilisez une base de données relationnelle, une astuce utile est d'utiliser la clause *where* en configurant les modules PAM. Par exemple, avec une base de données contenant les attributs de table suivants :

```
(user_id, user_name, realname, shell, password, UID, GID, homedir, sys, pop, ima
```

En modifiant les attributs de service en champs booléens, vous pouvez les utiliser pour permettre ou interdire l'accès aux différents services avec simplement les lignes appropriées dans les fichiers suivants :

- `/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"`.

Le système est vulnérable ! (En êtes-vous certain ?)

Le scanneur X de vérification des failles indique que le système Debian est vulnérable !

Plusieurs scanneurs de vérification de failles renvoient des faux positifs quand ils sont utilisés sur des systèmes Debian, car ils n'utilisent que le numéro de version pour déterminer si un paquet donné de logiciel est vulnérable, mais ils ne testent pas réellement la faille de sécurité elle-même. Comme Debian ne change de numéros de version lors de la correction d'un paquet (il est courant que la correction effectuée pour des versions plus récentes soit rétroportée), certains outils ont tendance à croire qu'un système Debian mis à jour est vulnérable alors qu'il ne l'est pas.

Si vous pensez que le système est à jour des correctifs de sécurité, vous pourriez utiliser les références croisées des bases de données des failles de sécurité publiées avec les DSA (consultez la section intitulée « Alertes de sécurité Debian ») pour éliminer les faux positifs, si l'outil utilisé inclut des références CVE.

Une attaque apparaît dans les fichiers journaux du système. Le système est-il compromis ?

Une trace d'une attaque ne veut pas toujours dire que le système a été compromis et vous devriez effectuer les étapes habituelles pour déterminer si le système est vraiment compromis (consultez Chapitre 11, *Après la compromission (la réponse à l'incident)*). Même si le système n'était pas vulnérable à l'attaque journalisée, un attaquant déterminé pourrait avoir utilisé une autre faille en plus de celles détectées.

D'étranges lignes « MARK » apparaissent dans les journaux : le système est-il compromis ?

Les lignes suivantes pourraient apparaître dans les fichiers journaux du système :

```
Dec 30 07:33:36 debian -- MARK --
Dec 30 07:53:36 debian -- MARK --
Dec 30 08:13:36 debian -- MARK --
```

Cela n'indique pas un type de compromission et les utilisateurs changeant de versions de Debian peuvent trouver cela étrange. Si le système n'a pas une charge importante (ou beaucoup de services actifs), ces lignes peuvent apparaître dans les journaux. C'est pour indiquer que le démon **syslogd** fonctionne correctement. De syslogd(8) :

```
-m interval
     Syslogd garde dans un journal une marque d'horodatage
     régulièrement. L'intervalle par défaut entre deux lignes
     -- MARK -- est de 20 minutes. Cela peut être modifié
     par cette option. Positionner l'intervalle à 0 le
     désactive complètement.
```

Des utilisateurs utilisant « su » apparaissent dans les journaux : le système est-il compromis ?

Vous pouvez trouver ce genre de lignes dans les journaux :

```
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 (
```

Ne vous inquiétez pas trop. Vérifiez que ces entrées sont dues à des tâches **cron** (habituellement, `/etc/cron.daily/find` ou **logrotate**) :

```
$ grep 25 /etc/crontab
25 9    * * *   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
```

« possible SYN flooding » apparaît dans les journaux : le système est-il attaqué ?

Si vous voyez ce genre d'entrées dans les fichiers journaux :

```
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.
```

Vérifiez le nombre de connexions au serveur en utilisant **netstat**, par exemple :

```
linux:~# netstat -ant | grep SYN_RECV | wc -l
   9000
```

Cela peut indiquer une attaque par déni de service (DoS) sur le port X du système (très certainement sur un service public comme un serveur web ou un serveur de courrier). Vous devriez activer TCP syncookies dans le noyau, consultez la section intitulée « Configurer syncookies ». Cependant, notez qu'une attaque par déni de service peut inonder le réseau même si vous pouvez l'empêcher de planter les systèmes (à cause de la raréfaction de descripteurs de fichiers, le système peut ne plus répondre avant que les connexions TCP expirent). Le seul moyen efficace pour arrêter cette attaque est de contacter le fournisseur d'accès réseau.

Des sessions superutilisateur étranges apparaissent dans les journaux : le système est-il compromis ?

Ce genre d'entrées peut apparaître dans le fichier `/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
```

Elles sont dues à l'exécution d'une tâche **cron** (dans cet exemple, toutes les cinq minutes). Pour déterminer le programme responsable de ces tâches, vérifiez les entrées dans : `/etc/crontab`, `/etc/cron.d`, `/etc/crond.daily` et la `crontab` du superutilisateur dans `/var/spool/cron/crontabs`.

Le système a été victime d'une intrusion, que faire ?

Plusieurs étapes sont à prendre en compte en cas d'intrusion.

- Vérifiez que le système est à jour avec les correctifs de sécurité pour les failles publiées. Si le système est vulnérable, les risques de compromission réelle du système augmentent. Les risques augmentent encore plus si la faille est connue depuis un certain temps, car il y a habituellement plus d'activité en lien avec d'anciennes failles. Voici un lien vers les http://www.sans.org/top20/.

- Consultez ce document, en particulier la section Chapitre 11, *Après la compromission (la réponse à l'incident)*.

- Demandez de l'aide. La liste de diffusion debian-security permet de demander conseil sur la manière de récupérer ou corriger le système.

- Informez le http://www.cert.org local (s'il existe, sinon vous pourriez contacter le CERT directement). Cela peut ou non vous aider, mais au minimum, cela informera le CERT des attaques en cours. Cette information est très précieuse pour déterminer les outils et attaques utilisés par la communauté *blackhat*.

Comment pister une attaque ?

En regardant les journaux (s'ils n'ont pas été modifiés), en utilisant un système de détection d'intrusions (consultez la section intitulée « Mise en place de détection d'intrusion »), **traceroute**, **whois** et outils similaires (y compris des analyses post-mortem) vous pourriez trouver la source de l'attaque. La réaction face à ces informations dépend uniquement des règles de sécurité, et de ce que *vous* considérez comme une attaque Un scan distant est-il une attaque ? Un test de failles de sécurité est-il une attaque ?

Le programme X dans Debian est vulnérable, que faire ?

Tout d'abord, vérifiez si la vulnérabilité a été annoncée sur les listes de diffusion publiques de sécurité (comme Bugtraq) ou autre forums. L'équipe de sécurité Debian se met à jour à l'aide de ces listes, elle peut donc déjà être consciente du problème. Ne lancez pas d'autres actions si l'annonce est sur http://security.debian.org.

Si rien n'a été publié, veuillez envoyer un message à propos des paquets concernés avec une description aussi détaillée que possible de la vulnérabilité rencontrée (la preuve par un code d'exploitation est aussi bienvenue) à mailto:team@security.debian.org. Cela vous mettra en rapport avec l'équipe de sécurité Debian.

Le numéro de version pour un paquet indique une version vulnérable !

Au lieu de mettre à jour vers une nouvelle version, Debian rétroporte le correctif de sécurité dans la version de la distribution stable. La raison d'agir ainsi est simple : cela permet d'assurer qu'une version a le moins de modifications possible, de cette manière les choses ne changeront pas ou ne se briseront pas à cause d'une mise à jour de sécurité. Vous pouvez vérifier qu'une version sécurisée du paquet est utilisée en regardant le journal de modifications du paquet ou en comparant le numéro exact de version (version amont et révision Debian) avec celui indiqué dans l'alerte de sécurité Debian (« Debian Security Advisory »).

Logiciels spécifiques

ProFTPD est vulnérable à une attaque de déni de service

Ajoutez `DenyFilter \*.*/` au fichier de configuration, pour plus d'informations, consultez http://www.proftpd.org/bugs.html.

Après l'installation de portsentry, de nombreux ports sont ouverts.

Il s'agit simplement du mode de fonctionnement de **portsentry**. Il ouvre environ 20 ports non utilisés pour tenter de détecter les scans de ports.

Questions concernant l'équipe de sécurité Debian

La Foire Aux Questions (FAQ) maintenue par l'équipe de sécurité Debian se trouve à l'adresse http://www.debian.org/security/faq. Veuillez consulter cette page web pour avoir des informations à jour.

Annexe A. Historique des versions

Historique des versions

Version 3-19.2	Sun May 19 2024	HolgerWansing<hwansing@mailbox.org>

Translation files synchronised with XML sources 3-19

Version 3-19.1	Mon May 1 2017	MarcosFouces<marcos.fouces@gmail.com>

Translation files synchronised with XML sources 3-19

Version 3-19	April 2017	MarcosFouces<marcos.fouces@gmail.com>

Migration vers Docbook XML.
Construction avec Publican. Plus d'utilisation d'un Makefile personnalisé.
Migrate svn repository to git.
Import chinese, italian, spanish, portuguese, japanese, russian, french and german translations to PO format.

Version 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

Version 3-17	January 2015	ThijsKinkhorst<thijs@debian.org>

Suppression de la mention des mots de passe « shadow » MD5.
Pas de recommandation de dselect pour bloquer les paquets.
Plus d'inclusion in extenso de la FAQ de l'équipe en charge de la sécurité parce qu'elle reproduit des informations documentées ailleurs et se trouve donc perpétuellement dépassée.
Section « mise à jour » au redémarrage après la mise à niveau de la bibliothèque pour mentionner le paquet needrestart.
Langage genré évité. Correctif de Myriam.
Utilisation d'en-têtes LSB pour le script du pare-feu. Correctif de Dominic Walden.

Version 3-16	January 2013	JavierFernández-Sanguino Peña.<jfs@debian.org>

Indication que le document n'est pas mis à jour avec les dernières versions.
Mise à jour des liens vers l'emplacement actuel des sources.
Mise à jour des renseignements de sécurité pour les publications les plus récentes.
Lien vers des renseignements pour les développeurs sur les sources en ligne au lieu de les garder dans ce document pour éviter les doublons.
Extension des renseignements sur la sécurisation d'accès à la console, y compris la limitation des touches SysRq magiques.
Mise à jour des renseignements sur les modules PAM, y compris une façon de restreindre les connexions en console, l'utilisation de cracklib et l'utilisation de fonctionnalités disponibles dans /etc/pam.d/login.
Retrait des références aux variables obsolètes de /etc/login.defs.
Référence à certains modules PAM disponibles pour permettre l'authentification à deux facteurs, pour les administrateurs voulant arrêter de partager des mots de passe.
Correction de l'exemple de script en annexe.
Correction d'erreurs de référence.
Pointer vers le projet SourceForge de Basille au lieu du site bastille-unix.org car il ne répond plus.

Version 3-15	December 2010	JavierFernández-Sanguino Peña<jfs@debian.org>

Modification de la référence au site web de Log Analysis car il n'est plus disponible.

Version 3-14 March 2009 JavierFernández-Sanguino Peña<jfs@debian.org>
Modification de la section indiquant comment choisir un système de fichiers. ext3 est maintenant le système de fichiers par défaut.
Modification du nom des paquets relatifs à Enigmail pour correspondre aux modifications de nom introduites dans Debian.
Version 3-13 February 2008 JavierFernández-Sanguino Peña<jfs@debian.org>
Changement des URL pointant Bastille Linux vers www.Bastille-UNIX.org car le domaine a été http://bastille-linux.sourceforge.net/press-release-newname.html.
Correction des liens sur les vers Linux dénommés Ramen et Lion.
Utilisation de linux-image dans les exemples à la place de l'ancien paquet kernel-image.
Corrections typographiques indiquées par Francesco Poli.
Version 3-12 August 2007 JavierFernández-Sanguino Peña<jfs@debian.org>
Mise à jour des informations au sujet des mises à jour de sécurité. Abandon du texte parlant de Tiger. Inclusion d'informations sur les outils update-notifier et adept (pour les stations) ainsi que debsecan. Ajout de quelques liens vers d'autres outils disponibles.
Division des applications de pare-feu selon les utilisateurs cibles et ajout de fireflier à la liste des applications de pare-feu pour postes de travail.
Retrait des références à libsafe, un paquet retiré du dépôt de Debian (en janvier 2006).
Correction de l'emplacement du fichier de configuration de syslog. Merci à John Talbut.
Version 3-11 January 2007 JavierFernández-Sanguino Peña<jfs@debian.org>
Merci à Francesco Poli pour sa révision étendue du document.
Retrait de la plupart des références à la version Woody car elle n'est plus disponible dans le dépôt principal et que le suivi en sécurité n'est plus disponible pour celle-ci.
Description de la restriction des utilisateurs pour qu'ils ne puissent faire que des transferts de fichiers.
Ajout d'une note au sujet de la décision de déclassification de debian-private.
Mise à jour du lien sur les guides de gestion des incidents.
Ajout d'une note indiquant que les outils de développement (compilateurs, etc.) ne sont plus installés par défaut dans Etch.
Ajout d'une note indiquant que les outils de développement (compilateurs, etc.) ne sont plus installés par défaut dans Etch.
Correction des références sur le serveur maître de sécurité.
Ajout de références vers de la documentation supplémentaire d'apt sécurisé.
Amélioration de la description des signatures APT.
Mise en commentaire de points qui ne sont pas encore finalisés au sujet des clefs publiques des miroirs officiels.
Correction du nom de l'équipe de sécurité Debian Testing (Debian Testing Security Team).
Retrait d'une référence à Sarge dans un exemple.
Mise à jour de la section sur les antivirus : clamav est maintenant disponible depuis Etch. Mention de l'installateur pour f-prot.
Retrait de toutes les références à freeswan, car il est désuet.
Description des problèmes liés aux changements des règles de firewall à distance et quelques conseils en notes de bas de page.
Mise à jour des informations sur l'installation d'IDS, mentionner BASE et la nécessité de mettre en place une base de données d'audit.
Réécriture de la section « lancer bind par un utilisateur non superutilisateur » car cela ne s'applique plus à BIND 9. Retrait de la référence au script init.d car les configurations doivent être faites à l'aide de /etc/default/.
Retrait de la méthode désuète de mise en place des règles d'iptables, car Woody n'est plus maintenu.
Retrait du conseil à propos de LOG_UNKFAIL_ENAB. Il devrait être positionné à 'no' (la valeur par défaut).

Ajout de plus d'informations au sujet de la mise à jour du système avec les outils de station de travail (y compris update-notifier) et description de l'utilisation d'aptitude pour mettre le système à jour. Noter aussi que dselect est déprécié.
Mise à jour du contenu de la FAQ et retrait de paragraphes redondants.
Relecture et mise à jour de la section sur les analyses post mortem de malwares.
Retrait ou correction de quelques liens morts.
Correction de nombreuses erreurs typographiques et grammaticales mentionnées par Francesco Poli.

Version 3-10	November 2006	JavierFernández-Sanguino Peña<jfs@debian.org>

Ajout d'exemples d'utilisation de l'option rdepends d'**apt-cache** tel que suggéré par Ozer Sarilar.
Correction de l'emplacement du manuel de l'utilisateur de Squid après qu'Oskar Pearson (son responsable) nous ait informé de son déplacement.
Correction des informations au sujet d'umask. C'est dans logins.defs (et non pas limits.conf) que cela peut être configuré pour toutes les connexions. Préciser les valeurs par défaut de Debian et suggérer des valeurs plus restrictives pour les utilisateurs et le superutilisateur. Merci à Reinhard Tartler pour avoir détecté cette erreur.

Version 3-9	October 2006	JavierFernández-Sanguino Peña<jfs@debian.org>

Ajout d'informations sur le suivi des vulnérabilités de sécurité et ajout de références à propos du système de suivi en sécurité de Debian testing.
Ajout d'informations sur le suivi en sécurité pour Debian testing.
Correction d'un grand nombre d'erreurs typographiques à partir de correctifs fournis par Simon Brandmair.
Ajout d'une section rédigée par Max Attems sur la façon de désactiver la console de superutilisateur avec initramfs.
Retrait des références à queso.
Signalement dans l'introduction que testing est maintenant suivie par l'équipe de sécurité de Debian.

Version 3-8	July 2006	JavierFernández-Sanguino Peña<jfs@debian.org>

Réécriture de la mise en place de prisons (chroot) SSH pour clarifier les différentes options disponibles. Merci à Bruce Park avoir fait remarquer diverses erreurs dans cette annexe.
Correction des appels de **lsof** tel que suggéré par Christophe Sahut.
Inclusion des correctifs d'Uwe Hermann corrigeant plusieurs erreurs typographiques.
Correction d'une erreur typographique soulignée par Moritz Naumann dans une référence.

Version 3-7	April 2006	JavierFernández-Sanguino Peña<jfs@debian.org>

Ajout d'une section sur les meilleures techniques de sécurité recommandées aux développeurs de Debian.
Ajout de commentaires au script d'un pare-feu par WhiteGhost.

Version 3-6	March 2006	JavierFernández-Sanguino Peña<jfs@debian.org>

Inclusion de correctifs de Thomas Sjögren qui expliquent que `noexec` fonctionne avec les « nouveau » noyaux. Ajout d'informations à propos de la gestion des fichiers temporaires ainsi que des liens vers de la documentation externe.
Ajout d'un lien vers le site de Dan Farmer et Wietse Venema sur l'analyse post mortem, tel que suggéré par Freek Dijkstra. Ajout de quelques liens additionnels sur l'analyse post mortem.
Correction de l'URL du site italien du CERT. Merci à Christoph Auer.
Réutilisation des informations du wiki de Joey Hess sur apt sécurisé et insertion dans la section sur les infrastructures.
Révision des sections se référant à d'anciennes versions (Woody ou Potato).
Correction de quelques problèmes esthétiques avec les correctifs proposés par Simon Brandmair.
Inclusion des correctifs de Carlo Perassi : les extraits de code sur les ACL sont désuets, les correctifs pour Openwall sont également désuets. Retrait des notes FIXME à propos des noyaux 2.2 et 2.4, hap est désuet (et absent du WNPP), retrait des références à Immunix (StackGuard appartient maintenant à Novell) et résolution d'un FIXME à propos de l'utilisation de bsign et elfsign.

Mise à jour des références au site Internet de SELinux afin qu'elles pointent vers le wiki (présentement la source d'informations la plus à jour).
Ajout de balises de fichiers et utilisation plus constante de l'expression « somme MD5 » avec un correctif de Jens Seidel.
Correctifs de Joost van Baal améliorant les informations dans la section sur les pare-feu (lien vers le wiki au lieu de faire une liste de tous les paquets disponibles sur les pare-feu). Ferme le bogue nº 339865.
Révision de la FAQ sur les statistiques sur les vulnérabilités. Merci à Carlos Galisteo de Cabo d'avoir mentionné que l'information n'était plus à jour.
Citation d'extraits du Contrat social Debian 1.1 au lieu de 1.0, tel que suggéré par Francesco Poli.

Version 3-5	November 2005	JavierFernández-Sanguino Peña<jfs@debian.org>

Note sur la section SSH que le chroot ne fonctionnera pas si vous utilisez l'option nodev dans la partition et indication des derniers paquets ssh avec le correctif chroot, merci à Lutz Broedel d'avoir signalé ces problèmes.
Correction de faute de frappe remarquée par Marcos Roberto Greiner (md5sum devrait être sha1sum dans l'extrait de code).
Inclusion du correctif de Jens Seidel corrigeant un certain nombre de noms de paquets et de fautes de frappe.
Légère mise à jour de la section d'outils, suppression des outils plus disponibles et ajout de nouveaux outils.
Réécriture de parties de la section liée à l'endroit où trouver ce document et des formats disponibles (le site web fournit une version PDF). Note également sur le fait que les copies sur d'autres sites et les traductions peuvent être désuètes (la plupart des liens fournis par Google pour le manuel sur d'autres sites sont vraiment obsolètes).

Version 3-4	August-September 2005	JavierFernández-Sanguino Peña<jfs@debian.org>

Amélioration des renforcements de sécurité post-installation liés à la configuration du noyau pour la protection au niveau réseau avec un fichier sysctl.conf fourni par Will Moy.
Amélioration de la section gdm, grâce à Simon Brandmair.
Corrections de faute de frappe de Frédéric Bothamy et Simon Brandmair.
Améliorations des sections post-installation liées à la façon de générer les sommes MD5 (ou SHA-1) des binaires pour vérification périodique.
Mise à jour des sections post-installation concernant la configuration checksecurity (qui était obsolète).

Version 3-3	June 2005	JavierFernández-Sanguino Peña<jfs@debian.org>

Ajout d'un extrait de code pour utiliser grep-available pour générer la liste des paquets dépendant de Perl. Comme demandé dans le bogue nº 302470.
Réécriture de la section sur les services réseau (quels sont les services installés et comment les désactiver).
Ajout de plus d'informations sur la section de déploiement des pots de miel mentionnant des paquets Debian utiles.

Version 3-2	March 2005	JavierFernández-Sanguino Peña<jfs@debian.org>

Extension de la section sur les limites de la configuration de PAM.
Ajout d'informations sur la façon d'utiliser pam_chroot pour openSSH (basé sur le README de pam_chroot).
Correction de problèmes mineurs signalés par Dan Jacobson.
Mise à jour des informations sur les correctifs du noyau basées sur un correctif de Carlo Perassi et également en ajoutant des notes sur les programmes obsolètes et les nouveaux correctifs de noyau disponibles (Adamantix).
Inclusion d'un correctif de Simon Brandmair qui corrige une phrase liée aux échecs de connexion dans un terminal.
Ajout de Mozilla/Thunderbird aux agents GPG valables comme suggéré par Kapolnai Richard.
Expansion de la section sur les mises à jour de sécurité en mentionnant les mises à jour de bibliothèques et de noyau et sur la façon de détecter quand les services doivent être redémarrés.

Réécriture de la section sur les pare-feu, déplacement vers le bas des informations qui s'appliquent à *Woody* et expansion des autres sections incluant des informations sur la façon de mettre en place manuellement le pare-feu (avec un exemple de script) et sur la façon de tester la configuration du pare-feu.
Ajout d'informations préparatoires pour la version 3.1 de Debian.
Ajout d'informations plus détaillées sur les mises à jour du noyau, particulièrement destinées à ceux qui ont utilisé l'ancien système d'installation.
Ajout d'une petite section sur la version 0.6 d'apt expérimentale qui fournit des vérifications de signature de paquets. Déplacement de l'ancien contenu dans la section et également ajout d'un pointeur vers les changements réalisés dans aptitude.
Corrections de fautes de frappe signalées par Frédéric Bothamy.

Version 3-1 January 2005 JavierFernández-Sanguino Peña<jfs@debian.org>

Ajout de clarification sur /usr en lecture seule avec un correctif de Joost van Baal.
Application d'un correctif de Jens Seidel corrigeant plusieurs fautes de frappe.
FreeSWAN est mort, longue vie à OpenSWAN.
Ajout d'informations sur la restriction d'accès aux services RPC (quand ils ne peuvent pas être désactivés), également inclusion d'un correctif fourni par Aarre Laakso.
Mise à jour du script apt-check-sigs d'aj.
Application du correctif de Carlo Perassi corrigeant des URL.
Application du correctif de Davor Ocelic corrigeant beaucoup d'erreurs, de fautes de frappe, URL, erreurs de grammaire et FIXME. Ajout également de plusieurs informations supplémentaires pour certaines sections.
Réécriture de la section sur l'audit utilisateur, mise en évidence de l'utilisation de script qui n'a pas certains des problèmes associés à l'historique du shell.

Version 3-0 December 2004 JavierFernández-Sanguino Peña<jfs@debian.org>

Réécriture des informations sur l'audit utilisateur et inclusion d'exemples sur la façon d'utiliser script.

Version 2-99 March 2004 JavierFernández-Sanguino Peña<jfs@debian.org>

Ajout d'informations sur des références dans la compatibilité entre DSA et CVE.
Ajout d'informations sur apt 0.6 (apt sécurisé intégré dans experimental).
Correction de l'emplacement du HOWTO Chroot des démons comme suggéré par Shuying Wang.
Modification de la ligne APACHECTL dans l'exemple de chroot Apache (même si elle n'est pas du tout utilisé) comme suggéré par Leonard Norrgard.
Ajout d'une note concernant les attaques de liens directs (« hardlink ») si les partitions ne sont pas mises en place correctement.
Ajout de certaines étapes manquantes pour exécuter bind comme named tel que fourni par Jeffrey Prosa.
Ajout de notes à propos de l'obsolescence de Nessus et de Snort dans Woody et disponibilité de paquets rétroportés.
Ajout d'un chapitre concernant des vérifications de test d'intégrité périodiques.
Clarification de l'état de testing concernant les mises à jour de sécurité. (bogue Debian nº 233955).
Ajout d'informations concernant les contenus attendus dans securetty (comme c'est spécifique au noyau).
Ajout de pointeur pour snoopylogger (bogue Debian nº 179409).
Ajout d'une référence sur guarddog (bogue Debian nº 170710).
apt-ftparchive est dans apt-utils, pas dans apt (merci à Emmanuel Chantreau pour l'avoir signalé).
Suppression de jvirus de la liste des antivirus.

Version 2-98 JavierFernández-Sanguino Peña<jfs@debian.org>

Correction de l'URL comme suggéré par Frank Lichtenheld.
Correction d'une faute de frappe PermitRootLogin comme suggéré par Stefan Lindenau.

Version 2-97 September 2003 JavierFernández-Sanguino Peña<jfs@debian.org>

Ajout des personnes qui ont contribué significativement à ce manuel (merci de m'envoyer un message si vous pensez que vous devriez être dans la liste et que vous n'y êtes pas).

Ajout de quelques bla-bla à propos des FIXME/TODO.
Déplacement des informations sur les mises à jour de sécurité au début de la section comme suggéré par Elliott Mitchell.
Ajout de grsecurity à la liste des kernel-patches pour la sécurité, mais ajout d'une note sur les problèmes actuels avec celui-ci comme suggéré par Elliott Mitchell.
Suppression de boucles (echo to 'all') dans le script de sécurité réseau du noyau comme suggéré par Elliott Mitchell.
Ajout de plus d'informations (à jour) dans la section antivirus.
Réécriture de la section de protection des dépassements de tampon et ajout de plus d'informations sur les correctifs pour le compilateur pour activer ce type de protection.

Version 2-96	August 2003	JavierFernández-Sanguino Peña<jfs@debian.org>

Suppression (et nouvel ajout) de l'annexe sur Apache dans un chroot. L'annexe est maintenant sous une double licence.

Version 2-95	June 2003	JavierFernández-Sanguino Peña<jfs@debian.org>

Corrections de fautes signalées par Leonard Norrgard.
Ajout d'une section sur la façon de contacter le CERT pour la gestion d'incident (Chapitre 11, *Après la compromission (la réponse à l'incident)*).
Plus d'informations sur la mise en place d'un serveur mandataire (« proxy ») Squid.
Ajout d'un pointeur et suppression d'un FIXME grâce à Helge H. F.
Correction d'une faute (save_inactive) signalée par Philippe Faes.
Corrections de plusieurs fautes signalées par Jaime Robles.

Version 2-94	April 2003	JavierFernández-Sanguino Peña<jfs@debian.org>

Selon les suggestions de Maciej Stachura, j'ai développé la section sur les limitations pour les utilisateurs.
Correction d'une faute signalée par Wolfgang Nolte.
Correction de liens avec un correctif fourni par Ruben Leote Mendes.
Ajout d'un lien vers l'excellent document de David Wheeler dans la note sur le décompte des failles de sécurité.

Version 2-93	March 2003	FrédéricSchütz<schutz@math-gen.ch>

Réécriture complète de la section sur les attributs ext2 (lsattr/chattr).

Version 2-92	February 2003	JavierFernández-Sanguino Peña<jfs@debian.org>, FrédéricSchütz<schutz@math-gen.ch>

Fusion de la section 9.3 (« correctifs noyau utiles ») dans la section 4.13 (« Ajouter des correctifs noyau ») et ajout d'un peu de contenu.
Ajout de quelques TODO supplémentaires.
Ajout d'informations sur la façon de vérifier manuellement les mises à jour et également sur cron-apt. Ainsi Tiger n'est plus vu comme le seul moyen de faire des vérifications de mises à jour automatiques.
Légère réécriture de la section sur l'exécution des mises à jour de sécurité grâce aux commentaires de Jean-Marc Ranger.
Ajout d'une note sur l'installation de Debian (qui suggérera à l'utilisateur une mise à jour de sécurité juste après l'installation).

Version 2-91	January/February 2003	JavierFernández-Sanguino Peña<jfs@debian.org>

Ajout d'un correctif proposé par Frédéric Schütz.
Ajout de quelques références supplémentaires sur les capacités grâce à Frédéric.
Modifications légères sur la section bind par l'ajout d'une référence à la documentation en ligne de BIND 9 et de références correctes dans la première zone (Salut Pedro !)
Correction de la date du journal de modifications – nouvelle année :-).
Ajout d'une référence aux articles de Colin pour les TODO.

Suppression de la référence à d'anciens correctifs SSH+chroot.
Correctifs additionnels de Carlo Perassi.
Corrections de fautes (récursif dans BIND est récursion) signalées par Maik Holtkamp.

Version 2-9 December 2002 JavierFernández-Sanguino Peña<jfs@debian.org>

Réorganisation des informations sur chroot (fusion de deux sections, cela n'avait pas de sens de les garder séparées).
Ajout de notes sur le chroot d'Apache fournies par Alexandre Ratti.
Application de correctifs proposés par Guillermo Jover.

Version 2-8 JavierFernández-Sanguino Peña<jfs@debian.org>

Application des correctifs de Carlo Perassi, corrections incluant : modification de la longueur de lignes, correction d'URL, et correction de certains FIXME.
Mise à jour du contenu de la FAQ de l'équipe en charge de la sécurité de Debian.
Ajout d'un lien vers la FAQ de l'équipe en charge de la sécurité de Debian et la référence du développeur Debian, les sections dupliquées pourraient (juste pourraient) être supprimées à l'avenir.
Correction de la section d'audit manuel avec les commentaires de Michal Zielinski.
Ajout d'un lien vers des dictionnaires (contribution de Carlo Perassi).
Correction de quelques erreurs de frappe (il en reste encore plein).
Correction des liens TDP conformément à la suggestion de John Summerfield.

Version 2-7 JavierFernández-Sanguino Peña<jfs@debian.org>

Correction de quelques fautes qui ont été signalées par Tuyen Dinh, Bartek Golenko et Daniel K. Gebhart.
Note concernant les rootkits utilisant /dev/kmem suggérée par Laurent Bonnaud.
Correction de fautes et de FIXME par Carlo Perassi.

Version 2-6 September 2002 CrisTillman<tillman@voice-trak.com>

Modifications pour améliorer la grammaire et l'orthographe.
s/host.deny/hosts.deny/ (1 endroit).
Application du correctif de Larry Holish (assez gros, corrige de nombreux FIXME).

Version 2-5.1 September 2002 JavierFernández-Sanguino Peña<jfs@debian.org>

Corrections de quelques fautes signalées par Thiemo Nagel.
Ajout d'une note de bas de page sur les conseils de Thiemo Nagel.
Corrige une URL.

Version 2-5.0 August 2002 JavierFernández-Sanguino Peña<jfs@debian.org>

Application d'un correctif fourni par Philipe Gaspar concernant Squid qui supprime aussi un FIXME.
Encore une autre FAQ concernant les bannières de services provenant de la liste de diffusion debian-security (discussion « Telnet information » démarrée le 26 juillet 2002).
Ajout d'une note concernant l'utilisation des références croisées CVE dans l'élément de la FAQ *En combien de temps l'équipe en charge de la sécurité de Debian...*
Ajout d'une nouvelle section concernant les attaques ARP fournie par Arnaud « Arhuman » Assad.
Nouvelle FAQ concernant dmesg et le démarrage en mode console par le noyau.
Petites parcelles d'informations sur les problèmes de vérification de signature dans les paquets (il semble qu'ils n'aient pas passé le stade de la version bêta).
Nouvelle FAQ concernant les faux positifs des outils d'évaluation de vulnérabilité.
Ajout de nouvelles sections au chapitre qui contient des informations sur les signatures de paquet et réorganisation en un nouveau chapitre *Infrastructure de sécurité Debian*.
Nouvel élément de FAQ concernant Debian et les autres distributions Linux.
Nouvelle section sur les clients de messagerie avec des fonctionnalités GPG/PGP dans le chapitre outils de sécurité.
Clarification sur la manière d'activer les mots de passe MD5 dans *Woody*, ajout d'un lien vers PAM ainsi qu'une note concernant la définition de max dans PAM.

Ajout d'une nouvelle annexe sur la façon de créer des environnements « chroot » (après avoir joué un peu avec makejail et avoir aussi corrigé quelques-uns de ses bogues), intégration des informations dupliquées dans toutes les annexes.
Ajout d'informations complémentaires concernant le « chrootage » de **SSH** et de son impact sur les transferts sécurisés de fichiers. Certaines informations ont été récupérées de la liste de diffusion debian-security (juin 2002 discussion : *Secure file transfers*).
Nouvelles sections sur la mise à jour automatique des systèmes Debian ainsi que les dangers d'utiliser la distribution « testing » ou la distribution « unstable » du point de vue des mises à jour de sécurité.
Nouvelle section, concernant la manière de rester à jour avec la mise en place de correctifs de sécurité, dans la section *avant la compromission* ainsi qu'une nouvelle section sur la liste de diffusion debian-security-announce.
Ajouts d'informations sur la manière de créer automatiquement des mots de passe sûrs.
Nouvelle section relative à la connexion des utilisateurs oisifs (*idle*).
Réorganisation de la section sécurisation du serveur de mail suite à la discussion *Secure/hardened/minimal Debian (ou « Why is the base system the way it is? » : pourquoi le système de base est-il comme ça ?)* sur la liste de diffusion debian-security (mai 2002).
Réorganisation de la section sur les paramètres réseau du noyau, avec les informations fournies par la liste de diffusion debian-security (mai 2002, discussion *syn flood attacked?*) et ajout d'un nouvel élément de FAQ.
Nouvelle section sur la manière de vérifier les mots de passe des utilisateurs et quels paquets utiliser pour cela.
Nouvelle section sur le chiffrement PPTP avec les clients Microsoft discuté sur la liste de diffusion debian-security (avril 2002).
Ajout d'une nouvelle section décrivant les problèmes qui peuvent survenir lorsque l'on attribue une adresse IP spécifique pour chaque service, cette information a été écrite d'après une discussion qui s'est tenue sur la liste de diffusion de Bugtraq : *Linux kernel 2.4 "weak end host" issue (discuté précédemment sur debian-security sous le titre « arp problem »)* (démarré le 9 mai 2002 par Felix von Leitner).
Ajout d'informations sur le protocole **SSH** version 2.
Ajout de deux sous-sections relatives à la configuration sécurisée d'Apache (c'est-à-dire, les éléments spécifiques à Debian).
Ajout d'une nouvelle FAQ traitant des « raw sockets », une relative à /root, une partie traitant des groupes d'utilisateurs et une autre traitant des permissions des journaux et des permissions des fichiers de configuration.
Ajout d'un lien vers un bogue dans libpam-cracklib qui pourrait encore être présent... (besoin de vérifier).
Ajout de plus d'informations sur l'analyse avancée (en attente de plus de renseignements sur les outils d'inspection de paquet tels que **tcpflow**).
Transformation de « Que dois-je faire concernant la compromission » en une série d'énumérations et en y ajoutant plus d'éléments.
Ajout d'informations sur la configuration de Xscreensaver pour verrouiller l'écran automatiquement après une durée donnée.
Ajout d'une note sur les utilitaires que vous ne devriez pas installer sur un système. Inclusion d'une note concernant Perl et pourquoi il ne peut pas être retiré facilement de Debian. L'idée vient de la lecture des documents d'Intersect concernant le renforcement de Linux.
Ajout d'informations sur lvm et les systèmes de fichiers journalisés, ext3 est préconisé. Les informations pourraient cependant y être trop génériques.
Ajout d'un lien sur la version texte disponible en ligne (à vérifier).
Ajout d'informations additionnelles sur la protection par pare-feu d'un système local, faisant suite à un commentaire d'Hubert Chan sur la liste de diffusion.
Ajout d'informations sur les limites de PAM et de liens vers les documents de Kurt Seifried (relatifs à un de ses messages sur Bugtraq le 4 avril 2002 répondant à une personne qui « découvrit » une vulnérabilité dans Debian GNU/Linux relative à l'insuffisance de ressources).
Comme suggéré par Julián Muñoz, ajout d'informations supplémentaires sur l'umask par défaut de Debian et ce à quoi un utilisateur peut accéder si on lui a donné une invite de commande sur le système (effrayant, non ?)

Inclusion d'une note dans la section du mot de passe BIOS suite à un commentaire d'Andreas Wohlfeld.
Inclusion des correctifs fournis par Alfred E. Heggestad corrigeant beaucoup de fautes encore présentes dans le document.
Ajout d'un lien vers le journal de modifications dans la section des remerciements car la plupart des personnes qui ont contribué sont cités ici (et pas là-bas).
Ajout de quelques notes complémentaires dans la section de chattr et d'une nouvelle section après l'installation qui parle des images systèmes. Les deux idées sont la contribution de Kurt Pomeroy.
Ajout d'une nouvelle section après l'installation juste pour rappeler aux utilisateurs de changer la séquence de démarrage.
Ajout d'éléments restant à faire (TODO) fournis par Korn Andras.
Ajout d'un lien vers les recommandations du NIST sur la manière de sécuriser un DNS. Cette contribution nous est fournie par Daniel Quinlan.
Ajout d'un petit paragraphe concernant l'infrastructure des certificats SSL de Debian.
Ajout des suggestions de Daniel Quinlan concernant l'authentification **SSH** et la configuration d'Exim en relais.
Ajout de plus d'informations concernant la sécurisation de BIND incluant les modifications suggérées par Daniel Quinlan et une annexe avec un script pour faire quelques uns des changements commentés dans cette section.
Ajout d'un lien vers un autre élément concernant le « chrootage » de BIND (a besoin d'être fusionné).
Ajout d'une ligne de Cristian Ionescu-Idbohrn pour récupérer les paquets avec gestion des tcpwrappers.
Ajout d'un peu plus d'informations sur la configuration PAM par défaut de la Debian.
Inclusion d'une question dans la FAQ au sujet de l'utilisation de PAM pour fournir des services sans compte shell.
Déplacement de deux éléments de la FAQ dans une autre section et ajout d'une nouvelle FAQ concernant la détection des attaques (et des systèmes corrompus).
Inclusion d'informations sur la configuration d'un pont pare-feu (incluant une annexe d'exemple). Merci à François Bayart qui m'a envoyé ça en mars.
Ajout d'une FAQ concernant les *MARK* d'heartbeat dans le syslogd d'après une question à laquelle Noah Meyerhans et Alain Tesio ont répondu en décembre 2001.
Inclusion d'informations sur la protection contre les débordements de tampons ainsi que quelques informations sur les correctifs du noyau.
Ajout d'informations supplémentaires (et réorganisation) de la section pare-feu. Mise à jour des informations concernant le paquet iptables et les générateurs de pare-feu disponibles.
Réorganisation des informations concernant la vérification des journaux, déplacement des informations de logcheck sur la détection d'intrusion machine vers cette section.
Ajout d'informations sur la manière de préparer un paquet statique pour BIND dans l'optique d'un « chrootage » (non testé).
Ajout d'un élément de FAQ concernant certains serveurs/services spécifiques (pourrait être développé avec quelques unes des recommandations de la liste de diffusion debian-security).
Ajout d'informations sur les services RPC (et quand ils sont nécessaires).
Ajout de plus d'informations sur les capacités (« capabilities ») en matière de sécurité (et ce que fait lcap). Y a-t-il une bonne documentation sur ce sujet ? Je n'ai trouvé aucune documentation sur mon noyau 2.4.
Correction de fautes de frappes.

Version 2-4 June 2002 JavierFernández-Sanguino Peña<jfs@debian.org>

Réécriture d'une partie de la section BIOS.

Version 2-3.1 April 2002 JavierFernández-Sanguino Peña<jfs@debian.org>

Encadrement de la plupart des emplacements de fichiers par la balise « file ».
Correction de fautes signalées par Edi Stojicevi.
Légère modification de la section des outils d'audit distant.
Ajout d'éléments à faire.
Ajout d'informations concernant les imprimantes et du fichier de configuration de CUPS (tiré d'une discussion sur debian-security).

Ajout d'un correctif proposé par Jesus Climent concernant l'accès d'utilisateurs valables du système à ProFTPD quand il est configuré en serveur anonyme.
Petite modification aux schémas de partitionnement dans le cas particulier des serveurs de messagerie.
Ajout du livre « Hacking Linux Exposed » à la section des livres.
Correction d'une faute de frappe sur un répertoire signalée par Eduardo Pérez Ureta.
Correction d'une coquille dans /etc/ssh dans la liste de contrôle signalée par Edi Stojicevi.

Version 2-3.0	April 2002	JavierFernández-Sanguino Peña<jfs@debian.org>

Correction de l'emplacement du fichier de configuration de dpkg.
Suppression d'Alexander des informations sur les contacts.
Ajout d'une autre adresse électronique.
Correction de l'adresse électronique d'Alexander (même si elle est commentée).
Correction de l'emplacement des clefs de versions (merci à Pedro Zorzenon pour avoir relevé cette erreur).

Version 2-2	April 2002	JavierFernández-Sanguino Peña<jfs@debian.org>

Corrections de fautes, merci à Jamin W. Collins pour ces corrections.
Ajout d'une référence à la page de manuel d'apt-extracttemplate (documentation sur la configuration de APT::ExtractTemplate).
Ajout d'une section concernant la limitation de SSH. Informations basées sur celles qui ont été postées par Mark Janssen, Christian G. Warden et Emmanuel Lacour sur la liste de diffusion debian-security.
Ajout d'informations sur les logiciels antivirus.
Ajout d'une FAQ : journaux de su provenant du fait que cron fonctionne en tant que superutilisateur.

Version 2-1	April 2002	JavierFernández-Sanguino Peña<jfs@debian.org>

Modifications du « FIXME » de lshell, merci à Oohara Yuuma.
Ajout d'un paquet sXid et retrait du commentaire étant donné qu'il est disponible.
De nombreuses fautes relevées par Oohara Yuuma ont été corrigées.
ACID est maintenant disponible dans Debian (dans le paquet acidlab).
Liens de LinuxSecurity corrigés (merci à Dave Wreski de nous l'avoir signalé).

Version 2-0	March 2002	JavierFernández-Sanguino Peña<jfs@debian.org>

Transformation du HOWTO en Manuel (maintenant je peux dire RTFM).
Ajout d'informations concernant l'encapsulation TCP et Debian (maintenant plusieurs services sont compilés avec la prise en charge adéquate ; ainsi cela n'est plus un problème d'**inetd**).
Clarification des informations sur la désactivation des services pour la rendre plus cohérente (les informations RPC se réfèrent toujours à update-rc.d).
Ajout d'une petite note sur lprn.
Ajout de quelques renseignements sur les serveurs corrompus (toujours très approximatif).
Corrections des fautes signalées par Mark Bucciarelli.
Ajout d'étapes supplémentaires sur la récupération des mots de passe lorsque l'administrateur a paramétré paranoid-mode=on.
Ajout d'informations pour paramétrer paranoid-mode=on lorsque l'on se connecte en mode console.
Nouveau paragraphe pour présenter la configuration des services.
Réorganisation de la section *Après l'installation* afin de permettre une lecture plus aisée du document.
Informations sur la manière de paramétrer des pare-feu avec l'installation standard de Debian 3.0 (paquet iptables).
Petit paragraphe détaillant pourquoi l'installation par le réseau n'est pas une bonne idée et comment on peut l'éviter en utilisant les outils Debian.
Petit paragraphe sur un article de l'IEEE qui souligne l'importance d'une application rapide des correctifs.
Annexe sur la manière de paramétrer une machine snort Debian, basé sur ce que Vladimir a envoyé à la liste de diffusion debian-security (le 3 septembre 2001).
Information sur la manière dont est configurée logcheck dans Debian et comment il peut être utilisé pour paramétrer HIDS.
Informations sur les comptes utilisateurs et sur les analyses de profils.

Inclusion de la configuration de apt.conf pour un /usr en lecture seule ; copié à partir du courrier d'Olaf Meeuwissen envoyé à la liste de diffusion debian-security.
Nouvelle section sur le VPN qui contient quelques liens ainsi que les paquets disponibles dans Debian (besoin de contenu concernant l'installation de VPN et les problèmes spécifiques à Debian) basé sur les courriers de Jaroslaw Tabor et Samuli Suonpaa postés sur la liste de diffusion debian-security.
Petite note concernant quelques programmes pour construire automatiquement des prisons « chrootées ».
Nouveau sujet de FAQ concernant identd d'après une discussion sur la liste de diffusion debian-security (février 2002, commencé par Johannes Weiss).
Nouveau sujet de FAQ concernant **inetd** d'après une discussion sur la liste de diffusion debian-security (février 2002).
Note d'introduction sur rcconf dans la section « désactivation de services ».
Diverses approches concernant le LKM. Remerciements à Philipe Gaspar.
Ajouts de liens vers les documents du CERT et les ressources Couterpane.

Version 1-99 January 2002 JavierFernández-Sanguino Peña<jfs@debian.org>

Ajout d'un nouveau sujet de FAQ concernant le temps de réaction à avoir pour corriger les failles de sécurité.
Réorganisation des sections de la FAQ.
Début d'une section concernant les pare-feu dans Debian GNU/Linux (pourrait être un peu élargie).
Corrections de fautes signalées par Matt Kraai.
Correction sur les informations DNS.
Ajout d'informations sur whisker et nbtscan à la section audit.
Correction d'URL erronées.

Version 1-98 January 2002 JavierFernández-Sanguino Peña<jfs@debian.org>

Ajout d'une nouvelle section concernant l'utilisation de Debian GNU/Linux pour réaliser des audits.
Ajout de renseignements sur le démon finger d'après la liste de diffusion debian-security.

Version 1-97 January 2002 JavierFernández-Sanguino Peña<jfs@debian.org>

Correction du lien pour Linux Trustees.
Correction de fautes (correctifs d'Oohara Yuuma et Pedro Zorzenon).

Version 1-96 December 2001 JavierFernández-Sanguino Peña<jfs@debian.org>

Réorganisation de la section installation et suppression de services et ajout de nouvelles notes.
Ajout de quelques notes concernant l'utilisation d'outils tels que les outils de détection d'intrusion.
Ajout d'un chapitre concernant la signature de paquets.

Version 1-95 December 2001 JavierFernández-Sanguino Peña<jfs@debian.org>

Ajout de notes concernant la sécurité de Squid envoyées par Philipe Gaspar.
Correction de liens rootkit. Merci à Philipe Gaspar.

Version 1-94 November 2001 JavierFernández-Sanguino Peña<jfs@debian.org>

Ajout de quelques notes concernant Apache et Lpr/lpng.
Ajout d'informations concernant les partitions noexec et readonly.
Réécriture de la manière dont les utilisateurs peuvent aider aux problèmes liés à la sécurité Debian (sujet d'une FAQ).

Version 1-93 November 2001 JavierFernández-Sanguino Peña<jfs@debian.org>

Correction de l'emplacement du programme mail.
Ajout de nouveaux sujets à la FAQ.

Version 1-92 October 2001 JavierFernández-Sanguino Peña<jfs@debian.org>

Ajout d'une petite section sur la manière dont Debian s'occupe de la sécurité.
Clarification sur les mots de passe MD5 (merci à « rocky »).

Ajout d'informations concernant le renforcement de X par Stephen van Egmond.
Ajout de nouveaux sujets à la FAQ.

Version 1-91	October 2001	JavierFernández-Sanguino Peña<jfs@debian.org>

Ajout d'informations détaillées envoyées par Yotam Rubin.
Ajout de renseignements sur la manière de mettre en place un « honeynet » en utilisant Debian GNU/Linux.
Ajout de TODO supplémentaires.
Correction de nouvelles fautes (merci Yotam !).

Version 1-9	October 2001	JavierFernández-Sanguino Peña<jfs@debian.org>

Correction des « fautes d'orthographe » et nouvelles informations (contributions de Yotam Rubin).
Ajout de liens vers d'autres documents en ligne (et hors ligne) tous deux figurant dans la section (consultez la section intitulée « Être conscient des problèmes de sécurité »).
Ajout d'informations sur la configuration d'options de BIND pour restreindre l'accès au serveur DNS.
Ajout d'informations sur la consolidation automatique d'un système Debian (par référence aux paquets harden et bastille).
Suppression de quelques TODO terminés et ajout de nouveaux.

Version 1-8	October 2001	JavierFernández-Sanguino Peña<jfs@debian.org>

Ajout de la liste des utilisateurs et des groupes standards, donnée par Joey Hess à la liste de discussion debian-security.
Ajout d'informations sur les « rootkits » LKM (la section intitulée « Loadable Kernel Modules (LKM) ») avec la contribution de Philipe Gaspar.
Ajout d'informations sur ProFTPD avec la contribution d'Emmanuel Lacour.
Rajout de l'annexe « pense-bête » d'Era Eriksson.
Ajout de nouveaux TODO et retrait de ceux terminés.
Ajout manuel des correctifs d'Era car ils n'ont pas été inclus dans la version précédente.

Version 1-7	September 2001	JavierFernández-Sanguino Peña<jfs@debian.org>, EraEriksson<era@iki.fi>

Fautes de frappes et changements de formulation.
Changements mineurs de balises : supprimer les balises tt et les remplacer par les balises command/package.

Version 1-6	August 2001	JavierFernández-Sanguino Peña<jfs@debian.org>

Ajout d'un lien sur le document publié dans le DDP (devrait à terme remplacer l'original).
Démarrage d'une mini-FAQ (qui devrait être élargie) avec quelques questions récupérées depuis ma boite de réception.
Ajout d'informations générales concernant la sécurisation.
Ajout d'un paragraphe au sujet de la distribution de courriers locaux.
Ajout de quelques liens vers d'autres sources d'informations.
Ajout d'informations sur le service d'impression.
Ajout d'une liste de tâches sur le renforcement de la sécurité.
Réorganisation des informations sur NIS et RPC.
Ajout de quelques notes lors de la lecture de ce document sur mon nouveau Visor :).
Correction de certaines lignes mal formatées.
Correction de fautes de frappes.
Ajout d'une idée Géniale/Paranoïaque avec la contribution de Gaby Schilders.

Version 1-5	May 2001	JavierFernández-Sanguino Peña<jfs@debian.org>, JosipRodin<joy@debian.org>

Ajout de paragraphes concernant BIND et quelques FIXME.

Version 1-4	May 2001	JavierFernández-Sanguino Peña<jfs@debian.org>

Petit paragraphe sur la vérification des « setuid »
Différents nettoyages mineurs.
Découverte de la manière d'utiliser `sgml2txt -f` pour la version texte.

Version 1-3 March 2001 JavierFernández-Sanguino Peña<jfs@debian.org>

Ajout de mise à jour de sécurité après le paragraphe « après installation ».
Ajout d'un paragraphe ProFTPD.
Cette fois, quelque chose concernant XDM a réellement été écrit. Désolé pour la dernière fois.

Version 1-2 December 2000 JavierFernández-Sanguino Peña<jfs@debian.org>

Beaucoup de corrections grammaticales de James Treacy, nouveau paragraphe XDM.

Version 1-1 December 2000 JavierFernández-Sanguino Peña<jfs@debian.org>

Corrections de fautes de frappes, divers ajouts.

Version 1-0 December 2000 JavierFernández-Sanguino Peña<jfs@debian.org>

Première publication.

Annexe B. Annexe

La procédure de durcissement étape par étape

Vous trouverez ci-dessous une procédure post-installation pour durcir un système Debian 2.2 GNU/Linux. Il s'agit d'une approche possible pour une telle procédure et celle-ci est orientée sur le renforcement des services réseaux. Elle est incluse pour présenter le processus entier que vous pouvez utiliser pendant la configuration. Veuillez également consultez la section intitulée « Liste des contrôles de configuration ».

- Faire une installation du système (tenez compte des informations dans ce manuel concernant le partitionnement). Après l'installation du système de base, allez dans l'installation personnalisée, ne sélectionnez pas de paquets par tâches (*task*). Sélectionnez les mots de passe cachés (*shadow*).

- Passer les paquets en revue avec **dselect** et retirer les paquets non nécessaires mais sélectionnés auparavant avant de faire [I]nstall. Laisser le strict minimum de logiciels sur le système.

- Actualiser tous les logiciels à partir des paquets les plus récents disponibles sur security.debian.org comme décrit précédemment dans la section intitulée « Faire une mise à jour de sécurité ».

- Appliquer les suggestions présentées dans ce manuel concernant les quotas par utilisateur, les définitions des connexions et **lilo**.

- Faire une liste de services actifs sur le système. Exécuter ceci :

  ```
  $ ps aux
  $ netstat -pn -l -A inet
  # /usr/sbin/lsof -i | grep LISTEN
  ```

  Vous devrez installer lsof-2.2 pour que la troisième commande fonctionne (à exécuter en tant que superutilisateur). Vous devriez faire attention car **lsof** peut traduire le mot LISTEN en fonction des paramètres régionaux.

- Afin de retirer les services non nécessaires, déterminer avant tout les paquets fournissant ces services et la façon de les démarrer. Cette tâche peut être facilement réalisée en vérifiant le programme qui écoute la « socket », l'exemple suivant le montre en utilisant ces outils et **dpkg** :

  ```
  #!/bin/sh
  # FIXME : c'est du vite fait, mal fait ; à remplacer par un bout
  # de script plus robuste
  for i in `sudo lsof -i | grep LISTEN | cut -d " " -f 1 |sort -u` ; do
    pack=`dpkg -S $i |grep bin |cut -f 1 -d : | uniq`
    echo "Le service $i est installé par $pack";
    init=`dpkg -L $pack |grep init.d/ `
    if [ ! -z "$init" ]; then
      echo "et démarré par $init"
    fi
  done
  ```

- Une fois les services indésirables trouvés, supprimer le paquet (avec **dpkg --purge**) ou utiliser **update-rc.d** (consultez la section intitulée « Désactivation de services démon ») de façon à le retirer du système de démarrage.

- Pour les services inetd (démarrés par le superdémon), vérifier les services activés dans `/etc/inetd.conf` avec :

```
$ grep -v "^#" /etc/inetd.conf | sort -u
```

  et désactiver ceux qui ne sont pas nécessaires en commentant la ligne qui les inclut dans `/etc/inetd.conf`, en supprimant le paquet ou en utilisant **update-inetd**.

- Si des services sont « encapsulés » (« wrapped ») (ceux utilisant **/usr/sbin/tcpd**), vérifier que les fichiers `/etc/hosts.allow` et `/etc/hosts.deny` sont configurés d'après les règles de services.

- Si le serveur utilise plus d'une interface externe, vous pourriez limiter les services pour n'en écouter qu'une seule. Par exemple, pour un accès FTP interne, paramétrez le démon FTP pour n'écouter que sur l'interface désirée et non toutes les interfaces (c'est-à-dire 0.0.0.0:21).

- Redémarrez la machine ou passez en mode utilisateur unique puis revenez en mode multiutilisateur avec :

```
# init 1
(...)
# init 2
```

- Vérifiez que les services sont maintenant disponibles et, si nécessaire, répétez les étapes ci-dessus.

- Installez maintenant les services nécessaires si ce n'est pas encore fait et configurez les correctement.

- Utilisez la commande d'interpréteur suivante pour déterminer l'utilisateur utilisé pour exécuter chaque service disponible :

```
# 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 "Le service $i a été démarré en tant qu'utilisateur $user"; done
```

  Pensez à modifier les utilisateur et groupe lançant ces services pour un couple utilisateur et groupe donné, et utilisez éventuellement **chroot** pour augmenter le niveau de sécurité. Vous pouvez procéder en changeant les scripts de démarrage de services de `/etc/init.d`. La plupart des services dans Debian utilisent **start-stop-daemon** qui propose des options (`--change-uid` et `--chroot`) pour faire cela. Un petit avertissement concernant l'utilisation de **chroot** pour des services est nécessaire : tous les fichiers installés par le paquet (consultez la sortie de dpkg -L) fournissant le service ainsi que les paquets dont il dépend peuvent être nécessaires dans l'environnement **chroot**. Des renseignements sur la mise en place d'un environnement **chroot** pour le programme **ssh** sont disponibles en la section intitulée « Environnement de chroot pour SSH ».

- Répéter les étapes ci-dessus afin de vérifier que seuls les services désirés sont en cours d'exécution et qu'ils fonctionnent avec une combinaison utilisateur et groupe désirée.

- Tester les services installés afin de voir si leur fonctionnement est bien celui souhaité.

- Vérifier le système en utilisant un scanner de vulnérabilités (comme nessus) de façon à déterminer les vulnérabilités du système (mauvaise configuration, services vieux ou non nécessaires).

- Mettre en place des mesures contre les intrusions de réseau et d'hôte comme snort et logcheck.

- Répéter l'étape du scanner de réseau et vérifier que le système de détection d'intrusion fonctionne correctement.

Pour les personnes vraiment paranoïaques, les considérations suivantes sont à envisager.

- Ajouter au système des possibilités de pare-feu, acceptant les connexions entrantes uniquement pour les services définis et limitant les connexions sortantes à celles autorisées.
- Revérifier l'installation avec une nouvelle évaluation de vulnérabilité à l'aide d'un scanner de réseaux.
- Vérifier les connexions sortantes en utilisant un scanner de réseaux depuis le système jusqu'à un hôte à l'extérieur et vérifier que les connexions non voulues ne trouvent pas leur sortie.

FIXME : Cette procédure considère le durcissement de service, mais pas le renforcement du système au niveau utilisateur, incluant des informations à propos de la vérification des droits d'utilisateurs, les fichiers setuid et le gel des changements dans le système en utilisant le système de fichiers ext2.

Liste des contrôles de configuration

Cette annexe récapitule brièvement les points des autres sections de ce manuel sous une forme condensée de liste de contrôles. C'est un petit résumé pour ceux qui ont déjà lu le manuel. D'autres listes de contrôles sont disponibles, y compris la http://seifried.org/security/os/linux/20020324-securing-linux-step-by-step.html de Kurt Seifried et la http://www.cert.org/tech_tips/usc20_full.html.

FIXME : C'est basé sur la version 1.4 du manuel et a peut-être besoin d'une mise à jour.

- Limiter les accès physiques et les possibilités de démarrage.
  - Activer un mot de passe pour le BIOS.
  - Désactiver le démarrage depuis disquette, CD, etc. dans le BIOS du système.
  - Mettre un mot de passe à LILO ou GRUB (respectivement `/etc/lilo.conf` ou `/boot/grub/menu.lst`) ; vérifier que le fichier de configuration de LILO ou de GRUB est en lecture seule.
- Partitionnement.
  - Séparer les données que les utilisateurs peuvent écrire, les données non système et les données d'exécution qui changent rapidement dans leurs propres partitions.
  - Mettre les options de montage `nosuid,noexec,nodev` dans `/etc/fstab` pour les partitions ext2 ou ext3 telles que `/home` ou `/tmp`.
- Hygiène pour les mots de passe et la sécurité des connexions.
  - Choisir un bon mot de passe pour le superutilisateur.
  - Installer et utiliser PAM.
    - Ajouter la prise en charge de MD5 à PAM et s'assurer (de manière générale) que les entrées dans les fichiers `/etc/pam.d/` qui autorisent l'accès à la machine ont un second champ dans le fichier pam.d positionné à `requisite` ou `required`.
    - Modifier `/etc/pam.d/login` pour que seul le superutilisateur puisse se connecter localement.
    - Indiquer également les consoles (*ttys*) autorisées dans `/etc/security/access.conf` et configurer généralement ce fichier pour limiter au maximum les connexions du superutilisateur.

    - Ajouter pam_limits.so pour définir des limites par utilisateur.
    - Modifier `/etc/pam.d/passwd` : augmenter la taille minimale du mot de passe (6 caractères par exemple) et activer MD5.
    - Ajouter éventuellement le groupe wheel à `/etc/group` ; ajouter l'entrée pam_wheel.so group=wheel au fichier `/etc/pam.d/su`.
    - Pour les contrôles personnalisés par utilisateur, utiliser les entrées appropriées de pam_listfile.so.
    - Avoir un fichier `/etc/pam.d/other` et mettre en place une sécurité resserrée.
  - Définir des limites dans `/etc/security/limits.conf` (remarquez que `/etc/limits` n'est pas utilisé si vous utilisez PAM).
  - Resserrer `/etc/login.defs` ; de même, si vous activez MD5 ou PAM, assurez-vous de faire également les modifications dans ce fichier.
  - Resserrer `/etc/pam.d/login`
  - Désactiver l'accès FTP au superutilisateur dans le fichier `/etc/ftpusers`.
  - Désactiver la connexion réseau du superutilisateur ; utiliser su(1) ou sudo(1) (considérer l'installation du paquet sudo).
  - Utiliser PAM pour imposer des contraintes supplémentaires sur les connexions ?
- Autres problèmes locaux de sécurité.
  - Modifications du noyau (consultez la section intitulée « Configuration des options réseau du noyau »).
  - Correctifs du noyau (consultez la section intitulée « Les utilitaires pour ajouter des correctifs au noyau »).
  - Resserrer les permissions sur les fichiers journaux (`/var/log/{last,fail}log`, journaux d'Apache).
  - Vérifier que la vérification de setuid est activée dans `/etc/checksecurity.conf`.
  - Penser à créer des fichiers journaux avec uniquement le droit d'ajout et des fichiers de configuration invariants en utilisant chattr (systèmes de fichiers ext2 ou ext3 uniquement).
  - Mettre en place une vérification d'intégrité des fichiers (consultez la section intitulée « Vérifier l'intégrité des systèmes de fichiers »). Installer debsums.
  - Impression de tous les fichiers journaux sur une imprimante locale ?
  - Graver la configuration sur un CD amorçable et démarrer dessus ?
  - Désactiver les modules pour le noyau ?
- Restreindre les accès réseaux.
  - Installer et configurer **ssh** (considérer « PermitRootLogin No » dans `/etc/ssh` et « PermitEmptyPasswords No » ; d'autres suggestions sont également dans le texte).
  - Désactiver ou supprimer **in.telnetd** s'il est installé.

    - Généralement, désactiver les services inutiles dans le fichier `/etc/inetd.conf` en utilisant `update-inetd --disable` (ou désactiver inetd complètement, ou utiliser une solution de rechange comme **xinetd** ou **rlinetd**).

    - Désactiver les autres services inutiles ; FTP, DNS, HTTP, etc. ne devraient pas être démarrés si vous n'en avez pas besoin et être surveillés régulièrement sinon. Dans la plupart des cas, les courriers électroniques devraient être fonctionnels, mais configurés uniquement pour la livraison locale.

    - Pour les services nécessaires, n'utilisez pas simplement les programmes usuels, recherchez des versions plus sécurisées disponibles dans Debian (ou depuis tout autre source). Peu importe celle choisie, assurez-vous de bien comprendre les risques induits.

    - Mettre en place des prisons **chroot** pour les utilisateurs et démons extérieurs.

    - Configurer un pare-feu et l'encapsulation TCP (consulter hosts_access(5)) ; considérer l'astuce pour `/etc/hosts.deny` dans le texte.

    - Si FTP est disponible, mettre en place un serveur FTP qui sera toujours démarré dans un environnement **chroot** dans le répertoire personnel de l'utilisateur.

    - Si X est disponible, désactiver l'authentification xhost et utiliser plutôt **ssh** ; de façon encore plus sécurisée, désactiver X à distance si possible (ajouter -nolisten tcp à la ligne de commande de X et désactiver XDMCP dans le fichier `/etc/X11/xdm/xdm-config` en affectant la valeur 0 à requestPort).

    - Désactiver l'accès distant aux imprimantes.

    - Chiffrer toute session IMAP ou POP par SSL ou **ssh** ; installer éventuellement stunnel pour fournir ce service aux utilisateurs de courrier à distance.

    - Mettre en place un hôte de journaux et configurer les autres machines pour qu'elles envoient les journaux à cet hôte (`/etc/syslog.conf`).

    - Sécuriser BIND, Sendmail et tout autre démon complexe (exécuter dans une prison **chroot** ; exécuter en tant que peudo-utilisateur non superutilisateur).

    - Installer tiger ou un outil similaire de détection d'intrusion réseau.

    - Installer snort ou un outil similaire de détection d'intrusion réseau.

    - Faire sans NIS et RPC si possible (désactiver portmap).

- Problèmes de règlement.

    - Expliquer aux utilisateurs les tenants et aboutissants des règles. Lorsque vous interdisez quelque chose habituellement disponible sur d'autres systèmes, fournissez-leur une documentation qui explique comment arriver aux mêmes résultats de façon plus sécurisée.

    - Interdire l'utilisation de protocoles qui utilisent des mots de passe en clair (**telnet**, **rsh** et similaire ; FTP, IMAP, HTTP, etc.)

    - Interdire les programmes qui utilisent la SVGAlib.

    - Utiliser les quotas de disque.

- Rester informé des problèmes de sécurité.

- S'abonner aux listes de discussions liées à la sécurité.
- Configurer apt pour les mises à jour de sécurité — ajouter une entrée (ou plusieurs entrées) à `/etc/apt/sources.list` pour http://security.debian.org/.
- Se rappeler périodiquement d'exécuter **apt-get update ; apt-get upgrade** (mettre en place peut-être une tâche **cron** ?) comme expliqué dans la section intitulée « Faire une mise à jour de sécurité ».

Paramétrage d'un IDS autonome

Un système Debian autonome peut être facilement configuré en tant que système de détection d'intrusion (IDS) avec snort et une interface web pour analyser les alertes de détection d'intrusion :

- installer un système de base Debian sans sélectionner de paquets supplémentaires ;
- installer une version de Snort avec prise en charge de base de données et configurer l'IDS pour journaliser les alertes dans la base de données ;
- télécharger et installer BASE (Basic Analysis and Security Engine) ou ACID (Analysis Console for Intrusion Databases). Le configurer pour utiliser la même base de données que Snort ;
- télécharger et installer les paquets nécessaires[1].

BASE est actuellement empaqueté pour Debian dans acidbase et ACID est empaqueté sous le nom d'acidlab[2]. Les deux paquets fournissent une interface web graphique à la sortie de Snort.

À part l'installation de base, vous aurez aussi besoin d'un serveur web (comme apache), un interpréteur **PHP** et une base de données relationnelle (comme postgresql ou mysql) où Snort enregistrera ses alertes.

Le système devrait être mis en place avec au moins deux interfaces : l'une connectée à un réseau de gestion (pour accéder aux résultats et maintenir le système), l'autre sans adresse IP liée au secteur du réseau à analyser. Le serveur web devrait être configuré pour n'écouter que sur l'interface connectée au réseau de gestion.

Les deux interfaces devraient être configurées dans le fichier de configuration standard Debian `/etc/network/interfaces`. Une adresse (sur le réseau de gestion) peut être configurée normalement. L'autre interface doit être configurée pour être démarrée lorsque le système démarre, mais sans adresse d'interface. La définition d'interface suivante peut être utilisée :

```
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

[1] Typiquement les paquets nécessaires seront installés par l'intermédiaire des dépendances.
[2] Il est aussi disponible au téléchargement depuis http://www.cert.org/kb/acid/, http://acidlab.sourceforge.net et http://www.andrew.cmu.edu/~rdanyliw/snort/.

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.

Configuration d'un pare-feu pont

Ces informations sont fournies par Francois Bayart pour aider les utilisateurs à mettre en place un pare-feu pont avec le noyau 2.4.x et iptables. Des correctifs de noyau ne sont plus nécessaires car le code est maintenant une partie standard de la distribution du noyau Linux.

Pour configurer le noyau avec la prise en charge nécessaire, exécutez `make menuconfig` ou `make xconfig`. Dans la section *Networking options*, activez les options suivantes :

```
[*] Network packet filtering (replaces ipchains)
[ ]   Network packet filtering debugging (NEW)
<*> 802.1d Ethernet Bridging
[*]   netfilter (firewalling) support (NEW)
```

Avertissement : vous devez désactiver ceci si vous voulez appliquer des règles de pare-feu ou sinon **iptables** ne fonctionnera pas :

```
[ ]   Network packet filtering debugging (NEW)
```

Ensuite, ajoutez les options correctes dans la section *IP: Netfilter Configuration*. Puis, compilez et installez le noyau. Si vous désirez le faire à la *sauce Debian*, installez kernel-package et exécutez **make-kpkg** pour créer un paquet noyau personnalisé Debian à installer sur le serveur en utilisant dpkg. Une fois le nouveau noyau compilé et installé, installez le paquet bridge-utils.

Une fois ces étapes achevées, vous pouvez terminer la configuration du pont. La section suivante présente deux configurations différentes possibles pour le pont, chacune avec une carte réseau hypothétique et les commandes nécessaires.

Un pont fournissant des fonctionnalités de traduction d'adresse (NAT) et de pare-feu

La première configuration utilise le pont comme un pare-feu avec traduction d'adresse réseau (NAT) qui protège un serveur et les clients du réseau interne. Voici ci-dessous un diagramme de la configuration du réseau :

```
Internet ---- routeur ---- pont
           (62.3.3.25)  (62.3.3.26 gw 62.3.3.25 / 192.168.0.1)
                          |
                          |
                          |---- serveur web
                          |   (62.3.3.27 gw 62.3.3.25)
                          |
                        réseau ---- Zipowz
                                  (192.168.0.2 gw 192.168.0.1)
```

Les commandes suivantes présentent une façon de configurer ce pont.

```
# Créer l'interface br0
/usr/sbin/brctl addbr br0

# Ajouter l'interface Ethernet à utiliser avec le pont
/usr/sbin/brctl addif br0 eth0
/usr/sbin/brctl addif br0 eth1

# Activer l'interface Ethernet
/sbin/ifconfig eth0 0.0.0.0
/sbin/ifconfig eth1 0.0.0.0

# Configurer le pont Ethernet
# Le pont sera correct et invisible (pare-feu transparent).
# Il est invisible à traceroute et la passerelle réelle est
# conservée sur les autres machines. La passerelle pourrait aussi
# être configurée sur le pont et être choisie comme nouvelle
# passerelle pour les autres machines.

/sbin/ifconfig br0 62.3.3.26 netmask 255.255.255.248 broadcast 62.3.3.31

# Cette IP interne est ajoutée pour créer la traduction d'adresse
ip addr add 192.168.0.1/24 dev br0
/sbin/route add default gw 62.3.3.25
```

Un pont fournissant des fonctionnalités de pare-feu

Une seconde possibilité est un système mis en place comme un pare-feu transparent pour un réseau avec un espace d'adresses IP publiques.

```
Internet ---- routeur ---- pont
            (62.3.3.25)  (62.3.3.26)
                              |
                              |
                              |---- serveur web
                              |   (62.3.3.28 gw 62.3.3.25)
                              |
                              |---- serveur de courriers
                                  (62.3.3.27 gw 62.3.3.25)
```

Les commandes suivantes présentent une façon de configurer ce pont.

```
# Créer l'interface br0
/usr/sbin/brctl addbr br0

# Ajouter l'interface Ethernet à utiliser avec le pont
/usr/sbin/brctl addif br0 eth0
/usr/sbin/brctl addif br0 eth1

# Activer l'interface Ethernet
```

```
/sbin/ifconfig eth0 0.0.0.0
/sbin/ifconfig eth1 0.0.0.0

# Configurer le pont Ethernet
# Le pont sera correct et invisible (pare-feu transparent).
# Il est invisible à traceroute et la passerelle réelle est
# conservée sur les autres machines. La passerelle pourrait aussi
# être configurée sur le pont et être choisie comme nouvelle
# passerelle pour les autres machines.

/sbin/ifconfig br0 62.3.3.26 netmask 255.255.255.248 broadcast 62.3.3.31
```

Si vous exécutez un traceroute vers le serveur de courriers Linux, vous ne verrez pas le pont. Si vous voulez accéder au pont avec **ssh**, vous devez utiliser une passerelle ou d'abord vous connecter sur un autre serveur comme le « serveur de courriers », puis ensuite vous connecter sur le pont par la carte réseau interne.

Règles de base d'iptables

Voici un exemple des règles de base qui pourraient être utilisées pour l'une ou l'autre des configurations.

Exemple B.1. Règles de base d'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 INVALID -j DROP
iptables -A FORWARD -m state --state ESTABLISHED,RELATED -j ACCEPT

# Quelques règles amusantes, mais pas pour un iptables classique,
# désolé...
# Limite ICMP
# iptables -A FORWARD -p icmp -m limit --limit 4/s -j ACCEPT
# Correspond à une chaîne de caractères, une bonne méthode simple pour
# bloquer certains VIRUS très rapidement
# iptables -I FORWARD -j DROP -p tcp -s 0.0.0.0/0 -m string \
                    --string "cmd.exe"

# Bloquer toutes les connexions MySQL simplement pour être sûr
iptables -A FORWARD -p tcp -s 0/0 -d 62.3.3.0/24 --dport 3306 -j DROP

# Règles du serveur de courriers Linux

# Autoriser 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

# Autoriser le serveur de courriers à se connecter à l'extérieur
# Remarque : ce n'est *pas* nécessaire pour les connexions précédentes
# (rappel : filtrage à état) et peut être supprimé.
iptables -A FORWARD -p tcp -s 62.3.3.27/32 -d 0/0 -j ACCEPT

# Règles pour le serveur WWW
```

```
# Autoriser les connexions HTTP (80) avec le serveur web
iptables -A FORWARD -p tcp -s 0.0.0.0/0 -d 62.3.3.28/32 --dport 80 \
                    -j ACCEPT

# Autoriser les connexions HTTPS (443) avec le serveur web
iptables -A FORWARD -p tcp -s 0.0.0.0/0 -d 62.3.3.28/32 --dport 443 \
                    -j ACCEPT

# Autoriser les connexions sortantes du serveur web
# Remarque : ce n'est *pas* nécessaire pour les connexions précédentes
# (rappel : filtrage à état) et peut être supprimé.
iptables -A FORWARD -p tcp -s 62.3.3.28/32 -d 0/0 -j ACCEPT
```

Exemple de script pour changer l'installation par défaut de BIND

Ce script automatise la procédure de modification d'installation par défaut du serveur de noms **bind** version 8 pour qu'il ne fonctionne *pas* en tant que superutilisateur. Remarquez que **bind** version 9 dans Debian fait déjà cela par défaut[3], et que vous devriez plutôt l'utiliser que **bind** version 8.

Ce script est laissé pour des raisons historiques et montre comment automatiser ce type de modifications globales du système. Le script créera les utilisateur et groupe définis pour le serveur de noms et modifiera à la fois `/etc/default/bind` et `/etc/init.d/bind` pour que le programme soit exécuté en tant que cet utilisateur. Utilisez-le avec la plus grande attention car il n'a pas été testé rigoureusement.

Vous pouvez aussi créer l'utilisateur vous-même et utiliser le correctif disponible pour le script d'initialisation par défaut attaché au http://bugs.debian.org/cgi-bin/bugreport.cgi?bug=157245.

```
#!/bin/sh
# Modifier la configuration par défaut du BIND v8 de Debian pour qu'il
# s'exécute en tant qu'utilisateur et groupe non superutilisateur.
#
# Ne pas utiliser cela avec la version 9, utiliser plutôt debconf pour le
# configurer.
#
# Attention : ce script n'a pas été testé rigoureusement, veuillez
# vérifier les modifications effectuées sur les scripts d'initialisation.

# (c) 2002 Javier Fernández-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.
#
```

[3] Depuis la version 9.2.1-5. C'est-à-dire depuis Debian *Sarge*.

```
#     Please see the file `COPYING' for the complete copyright notice.
#

restore() {
# Au cas où, restaurer le système si la modification échoue
  echo "Attention : restauration de la configuration précédente car il"
  echo "            est impossible de la modifier correctement."
  echo "Attention : veuillez vérifier le script $INITDERR."
  mv $INITD $INITDERR
  cp $INITDBAK $INITD
}


USER=named
GROUP=named
INITD=/etc/init.d/bind
DEFAULT=/etc/default/bind
INITDBAK=$INITD.preuserchange
INITDERR=$INITD.changeerror
AWKS="awk ' /\/usr\/sbin\/ndc reload/ { print \"stop; sleep 2; start;\"; noprint =

[ `id -u` -ne 0 ] && {
  echo "Ce script doit être exécuté en tant que superutilisateur"
  exit 1
}

RUNUSER=`ps eo user,fname |grep named |cut -f 1 -d " "`

if [ "$RUNUSER" = "$USER" ]
then
  echo "Attention : le démon de serveur de noms est déjà exécuté en tant"
  echo "            que $USER."
  echo "Erreur :    ce script ne modifiera pas la configuration."
  exit 1
fi
if [ ! -f "$INITD" ]
then
  echo "Erreur :    ce système n'a pas de $INITD (ce que ce script tente de"
  echo "            modifier)"
  RUNNING=`ps eo fname |grep named`
  [ -z "$RUNNING" ] && \
  echo "Erreur :    en fait, le démon de serveur de noms n'est même pas en"
  echo "            cours d'exécution (est-il installé ?)"
  echo "Erreur :    aucune modification ne sera apportée au système."
  exit 1
fi

# Vérifier si les options sont déjà configurées
if [ -e "$DEFAULT" ]
then
  if grep -q ^OPTIONS $DEFAULT; then
    echo "Erreur :    le fichier $DEFAULT a déjà des options configurées."
    echo "Erreur :    aucune modification ne sera apportée au système."
  fi
```

```
fi
# Vérifier si le groupe named existe
if [ -z "`grep $GROUP /etc/group`" ]
then
  echo "Création du groupe $GROUP :"
  addgroup $GROUP
else
  echo "Attention : le groupe $GROUP existe déjà. Il ne sera pas créé."
fi
# Pareil pour l'utilisateur
if [ -z "`grep $USER /etc/passwd`" ]
then
  echo "Création de l'utilisateur $USER :"
  adduser --system --home /home/$USER \
  --no-create-home --ingroup $GROUP \
  --disabled-password --disabled-login $USER
else
  echo "Attention : l'utilisateur $USER existe déjà. Il ne sera pas créé."
fi

# Modifier le script init.d

# D'abord faire une sauvegarde (vérifier qu'il n'y en a pas déjà une)
if [ ! -f $INITDBAK ]
then
  cp $INITD $INITDBAK
fi

# Puis l'utiliser pour la modifier
cat $INITDBAK |
eval $AWKS > $INITD

# Enfin placer les options dans le fichier /etc/default/bind
cat >>$DEFAULT <<EOF
# Utiliser l'utilisateur défini pour exécuter bind
OPTIONS="-u $USER -g $GROUP"
EOF
echo "Attention : le script $INITD a été modifié, tentative de test des"
echo "            modifications."
echo "Redémarrage du démon named (vérification des erreurs en cours)."

$INITD restart
if [ $? -ne 0 ]
then
  echo "Erreur :    échec du redémarrage du démon."
  restore
  exit 1
fi

RUNNING=`ps eo fname |grep named`
if [ -z "$RUNNING" ]
then
  echo "Erreur :    named n'est pas en cours d'exécution, c'est sans doute"
  echo "            dû à un problème avec les modifications."
```

```
  restore
  exit 1
 fi

# Vérifier que named fonctionne comme prévu
RUNUSER=`ps eo user,fname |grep named |cut -f 1 -d " "`

if [ "$RUNUSER" = "$USER" ]
then
  echo "Tout s'est bien passé, named semble maintenant fonctionner en tant"
  echo "            que $USER."
else
  echo "Erreur :    le script a échoué à modifier automatiquement le système."
  echo "Erreur :    named fonctionne actuellement en tant que $RUNUSER."
  restore
  exit 1
fi

exit 0
```

Le script précédent, exécuté sur le **bind** (version 8) personnalisé de Woody (Debian 3.0), modifiera le fichier initd après création de l'utilisateur et du groupe « named ».

Mise à jour de sécurité protégée par un pare-feu

Après une installation standard, un système peut toujours avoir des failles de sécurité. À moins de pouvoir télécharger les mises à jour pour les paquets vulnérables depuis un autre système (ou si vous avez fait un miroir de security.debian.org pour utilisation en local), le système devra être connecté à Internet pour les téléchargements.

Cependant, dès que vous vous connecter à Internet, vous exposez le système. Si l'un des services locaux est vulnérable, votre système peut même être compromis avant la fin de la mise à jour ! Cela peut sembler paranoïaque, mais une analyse du http://www.honeynet.org a démontré que les systèmes peuvent être compromis en moins de trois jours, même si le système n'est pas connu publiquement (c'est-à-dire, non publié dans les enregistrements DNS).

Lorsque vous faites une mise à jour sur un système non protégé par un système externe comme un pare-feu, il est possible de configurer correctement votre pare-feu pour restreindre les connexions n'impliquant que la mise à jour de sécurité elle-même. L'exemple ci-dessous montre comment mettre en place des telles fonctionnalités de pare-feu, ne permettant que les connexions à security.debian.org et en journalisant toutes les autres.

L'exemple suivant permet de configurer un jeu de règles de pare-feu restreint. Exécutez ces commandes depuis une console locale (pas à distance) pour limiter les risques de vous enfermer hors du système.

```
# 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 -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 -P INPUT DROP
# iptables -P FORWARD DROP
# iptables -P OUTPUT DROP
# 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,ESTABLIS
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
```

Remarque : l'utilisation d'une règle *DROP* dans la chaîne INPUT est la chose la plus correcte à faire, mais soyez *particulièrement* attentif lorsque c'est fait après avoir nettoyé la chaîne depuis une connexion distante. Lors d'un test des jeux de règles de pare-feu à distance, il est préférable d'exécuter un script avec le jeu de règles de pare-feu (au lieu d'introduire les règles une à une depuis la ligne de commande) et, par précaution, de garder une porte dérobée[4]

Bien sûr, toutes les portes dérobées devraient être désactivées avant de placer le système en production configurée pour pouvoir réactiver l'accès au système en cas d'erreur. Ainsi, il ne sera pas nécessaire de se déplacer pour corriger un jeu de règles de pare-feu bloquant.

FIXME : cela nécessite un DNS opérationnel puisqu'il est nécessaire pour résoudre security.debian.org. security.debian.org pourrait être ajouté à /etc/hosts mais c'est un nom canonique pour plusieurs hôtes (il y a plus d'un miroir de sécurité).

FIXME : cela ne fonctionnera qu'avec les URL HTTP car FTP peut avoir besoin du module ip_conntrack_ftp ou d'utiliser le mode passif.

Environnement de chroot pour SSH

Créer un environnement restreint pour **SSH** est un travail difficile à cause de ses dépendances et du fait que, à la différence d'autres serveurs, **SSH** fournit un interpréteur de commande distant pour les utilisateurs. C'est pourquoi vous devrez également considérer les applications que les utilisateurs auront le droit d'utiliser dans l'environnement.

[4] Par exemple *knockd*. Sinon, il est possible d'ouvrir une autre console et forcer le système à confirmer que quelqu'un est présent de l'autre côté, et réinitialiser la chaîne de pare-feu en absence de confirmation. Le script de test suivant pourrait servir :

```
#!/bin/bash while true; do read -n 1 -p "Y a-t-il quelqu'un ? " -t 30 ayt if [ -z "$ayt" ] ; then break fi done #
```

Deux options existent pour configurer une invite de commande à distance restreinte :

- chrooter les utilisateurs SSH, en configurant correctement le démon SSH pour chrooter un utilisateur après l'authentification juste avant de lui fournir une invite de commande. Chaque utilisateur peut avoir son propre environnement ;
- chrooter le serveur SSH, puisque l'application SSH est elle même chrootée, tous les utilisateurs sont chrootés dans l'environnement défini.

La première option a l'avantage de permettre d'avoir à la fois des utilisateurs chrootés ou non, en absence d'application setuid dans les chroots de l'utilisateur, il est plus difficile de s'en échapper. Cependant, vous pourriez configurer des chroots spécifiques à chaque utilisateur ce qui est plus délicat (car cela nécessite une coopération de la part du serveur SSH). La seconde option est plus facile à configurer, et protège d'une exploitation du serveur SH lui-même (puisqu'il est également dans le chroot) mais il sera limité de telle sorte que tous les utilisateurs partageront le même environnement chroot (impossible de configurer un environnement chroot par utilisateur).

Chrooter les utilisateur SSH

Le serveur SSH peut être configuré pour chrooter un ensemble d'utilisateurs définis dans une invite de commande possédant un jeu d'applications disponibles limité.

Utilisation de libpam-chroot

La façon probablement la plus facile est d'utiliser le paquet libpam_chroot fourni dans Debian. Une fois que vous l'avez installé, vous devez :

- modifier `/etc/pam.d/ssh` pour utiliser ce module PAM, ajouter cette ligne à la fin du fichier[5] :

```
session    required   pam_chroot.so
```

- configurer un environnement de chroot correct. Vous pouvez essayer d'utiliser les scripts disponibles en `/usr/share/doc/libpam-chroot/examples/`, utiliser le programme makejail[6] ou mettre en place un environnement Debian minimal avec debootstrap. Assurez-vous que l'environnement contient les périphériques nécessaires[7].
- configurer `/etc/security/chroot.conf` pour que les utilisateurs que vous déterminez soient chrootés dans le répertoire que vous avez mis en place auparavant. Vous pouvez vouloir ajouter des répertoires indépendants pour différents utilisateurs afin qu'ils ne puissent voir ni le système complet, ni les uns les autres ;

[5] L'option *debug* permet d'envoyer la progression du module à la facilité *authpriv.notice*.

[6] Vous pouvez créer un environnement bash très limité avc les définitions Python suivante pour makejail, en créant simplement le répertoire `/var/chroots/users/truc` et un fichier `bash.py` avec le contenu suivant :

```
chroot="/var/chroots/users/truc" cleanJailFirst=1 testCommandsInsideJail=["bash ls"]
```

Exécuter ensuite *makejail bash.py* pour créer l'environnement de l'utilisateur en `/var/chroots/users/truc`. Pour tester l'environnement, exécuter :

```
# chroot /var/chroots/users/truc/ ls bin dev etc lib proc sbin usr
```

;

[7] Dans certains cas, les périphériques `/dev/ptmx` et `/dev/pty*` et le sous-répertoire `/dev/pts/`. Exécuter MAKEDEV dans le répertoire `/dev` de l'environnement chrooté devrait suffire pour les créer s'ils n'existent pas. Avec les noyaux (version 2.6) qui créent dynamiquement les fichiers de périphérique, vous devrez créer les fichiers `/dev/pts/` vous-même et leur attribuer les droits nécessaires.

- configurer SSH : suivant la version d'OpenSSH, l'environnement chroot pourrait fonctionner directement sans effort, ou non. Depuis 3.6.1p2 la fonction *do_pam_session()* est appelée après que sshd ait abandonné ses droits, mais puisque chroot() a besoin des droits du superutilisateur, il ne fonctionnera pas avec la séparation de droits activés. Dans les versions plus récentes d'OpenSSH, cependant, le code PAM a été modifié et do_pam_session est appelé avant l'abandon des droits donc il fonctionnera même avec la séparation de droits activée. Si vous devez le désactiver, modifiez `/etc/ssh/sshd_config` ce cette façon :

  ```
  UsePrivilegeSeparation no
  ```

  Notez que cela réduira la sécurité de votre système car le serveur OpenSSH fonctionnera avec l'utilisateur *root*. Cela veut dire que si une attaque à distance est trouvée sur OpenSSH, un attaquant obtiendra les droits de *root* au lieu de ceux de *sshd*, ce qui compromettra le système en entier. [8]

Sans désactiver la *séparation de droits*, un `/etc/passwd` qui intègre l'UID de l'utilisateur dans le chroot sera nécessaire pour faire fonctionner correctement la *séparation de droits*.

Si la *séparation de droits* est définie à *yes* est que la version d'OpenSSH ne se comporte pas correctement, il faudra la désactiver. Si non, les utilisateurs qui essayent de se connecter au serveur et en étant chrootés par ce module verront ceci :

```
$ ssh -l user server
user@server's password:
Connection to server closed by remote host.
Connection to server closed.
```

C'est parce que le démon SSH, qui est exécuté en tant que « sshd », n'est pas capable de faire l'appel système chroot(). Pour désactiver la séparation de droits, il faut modifier les fichier de configuration `/etc/ssh/sshd_config` comme décrit précédemment.

Remarquez qu'en absence d'un des éléments suivants, les utilisateurs ne pourront pas se connecter au chroot :

- le système de fichiers `/proc` doit être monté dans le chroot des utilisateurs ;
- les périphériques `/dev/pts/` nécessaires doivent exister. Si les fichiers sont créés automatiquement par le noyau utilisé, il faut les créer vous-même dans le `/dev/` du chroot ;
- le répertoire personnel de l'utilisateur doit exister dans le chroot, sinon le démon SSH s'arrêtera.

Tous ses problèmes peuvent être débogués en utilisant le mot-clef *debug* dans la définition PAM de `/etc/pam.d/ssh`. En cas de problème, il peut être utile d'activer aussi le mode de débogage sur le client.

Note : ces renseignements sont également disponibles (et peut-être plus à jour) dans `/usr/share/doc/libpam-chroot/README.Debian.gz`, veuillez consulter ce fichier pour obtenir des renseignements à jour avant d'entreprendre les étapes ci-dessus.

Appliquer des correctifs au serveur SSH

Le serveur **sshd** de Debian ne vous autorisera pas à restreindre les mouvements des utilisateurs par le serveur étant donné que celui-ci est dépourvu de la fonction **chroot** que le programme commerciale **sshd2**

[8] Si vous utilisez un noyau implémentant le contrôle d'accès obligatoire (« Mandatory Access Control » ou MAC) (RSBAC/SElinux), vous pouvez éviter de changer cette configuration en autorisant simplement l'utilisateur *sshd* à exécuter l'appel système chroot().

possède (utilisation de « ChrootGroups » ou « ChrootUsers », consultez sshd2_config(5)). Toutefois, un correctif est disponible pour le faire sur le http://bugs.debian.org/139047. Il sera peut-être appliquée au paquet OpenSSH dans le futur. Emmanuel Lacour dispose de paquets Debian **ssh** pour *Sarge* avec cette fonctionnalité. Ils sont disponibles à http://debian.home-dn.net/sarge/ssh/. Notez que ceux-ci peuvent ne pas être à jour, effectuer l'étape de compilation est donc recommandé.

Après avoir appliqué le correctif, modifiez `/etc/passwd` en changeant le chemin du répertoire des utilisateurs (avec l'indicateur spécial `/./`) :

```
utilisateurjean:x:1099:1099:Jean Dupont Utilisateur:/home/michel/./:/bin/bash
```

Cela restreindra *à la fois* les accès distants au shell, ainsi que la copie par le tunnel **ssh**.

Assurez-vous que tous les programmes et bibliothèques sont bien présents dans le chemin **chroot**é pour les utilisateurs. Ces fichiers devraient appartenir à root pour éviter les fraudes de l'utilisateur (tel la sortie d'une prison **chroot**é). Un échantillon pourrait inclure ceci :

```
./bin:
total 660
drwxr-xr-x    2 root     root         4096 mars  18 13:36 .
drwxr-xr-x    8 guest    guest        4096 mars  15 16:53 ..
-r-xr-xr-x    1 root     root       531160 févr. 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 mars  18 13:36 more
-r-xr-xr-x    1 root     root         9916 juil. 26  2001 pwd
-r-xr-xr-x    1 root     root        24780 nov.  29 13:19 rm
lrwxrwxrwx    1 root     root            4 mars  30 16:29 sh -> bash


./etc:
total 24
drwxr-xr-x    2 root     root         4096 mars  15 16:13 .
drwxr-xr-x    8 guest    guest        4096 mars  15 16:53 ..
-rw-r--r--    1 root     root           54 mars  15 13:23 group
-rw-r--r--    1 root     root          428 mars  15 15:56 hosts
-rw-r--r--    1 root     root           44 mars  15 15:53 passwd
-rw-r--r--    1 root     root           52 mars  15 13:23 shells


./lib:
total 1848
drwxr-xr-x    2 root     root         4096 mars  18 13:37 .
drwxr-xr-x    8 guest    guest        4096 mars  15 16:53 ..
-rwxr-xr-x    1 root     root        92511 mars  15 12:49 ld-linux.so.2
-rwxr-xr-x    1 root     root      1170812 mars  15 12:49 libc.so.6
-rw-r--r--    1 root     root        20900 mars  15 13:01 libcrypt.so.1
-rw-r--r--    1 root     root         9436 mars  15 12:49 libdl.so.2
-rw-r--r--    1 root     root       248132 mars  15 12:48 libncurses.so.5
-rw-r--r--    1 root     root        71332 mars  15 13:00 libnsl.so.1
-rw-r--r--    1 root     root        34144 mars  15 16:10 libnss_files.so.2
-rw-r--r--    1 root     root        29420 mars  15 12:57 libpam.so.0
-rw-r--r--    1 root     root       105498 mars  15 12:51 libpthread.so.0
```

```
-rw-r--r--    1 root     root         25596 mars  15 12:51 librt.so.1
-rw-r--r--    1 root     root          7760 mars  15 12:59 libutil.so.1
-rw-r--r--    1 root     root         24328 mars  15 12:57 libwrap.so.0


./usr:
total 16
drwxr-xr-x    4 root     root          4096 mars  15 13:00 .
drwxr-xr-x    8 guest    guest         4096 mars  15 16:53 ..
drwxr-xr-x    2 root     root          4096 mars  15 15:55 bin
drwxr-xr-x    2 root     root          4096 mars  15 15:37 lib


./usr/bin:
total 340
drwxr-xr-x    2 root     root          4096 mars  15 15:55 .
drwxr-xr-x    4 root     root          4096 mars  15 13:00 ..
-rwxr-xr-x    1 root     root         10332 mars  15 15:55 env
-rwxr-xr-x    1 root     root         13052 mars  15 13:13 id
-r-xr-xr-x    1 root     root         25432 mars  15 12:40 scp
-rwxr-xr-x    1 root     root         43768 mars  15 15:15 sftp
-r-sr-xr-x    1 root     root        218456 mars  15 12:40 ssh
-rwxr-xr-x    1 root     root          9692 mars  15 13:17 tty


./usr/lib:
total 852
drwxr-xr-x    2 root     root          4096 mars  15 15:37 .
drwxr-xr-x    4 root     root          4096 mars  15 13:00 ..
-rw-r--r--    1 root     root        771088 mars  15 13:01 libcrypto.so.0.9.6
-rw-r--r--    1 root     root         54548 mars  15 13:00 libz.so.1
-rwxr-xr-x    1 root     root         23096 mars  15 15:37 sftp-server
```

Chrooter le serveur SSH

Si un chroot est créé pour inclure les fichiers du serveur SSH, par exemple `/var/chroot/ssh`, le serveur SSH **chroot**é serait démarré avec cette commande :

```
# chroot /var/chroot/ssh /sbin/sshd -f /etc/sshd_config
```

Cela ferait démarrer le démon **sshd** dans le chroot. Pour faire cela, il faut d'abord préparer le contenu du répertoire `/var/chroot/ssh` pour inclure à la fois le serveur SSH et tous les utilitaires dont les utilisateurs se connectant à ce serveur pourraient avoir besoin. Dans ce cas, vous devez vous assurer qu'OpenSSH utilise la *séparation de droits* (ce qui est le cas par défaut) en ayant la ligne suivante dans le fichier de configuration `/etc/ssh/sshd_config` :

```
UsePrivilegeSeparation yes
```

De cette façon, le démon distant fera aussi peu de choses que possible de même que le superutilisateur donc même en cas de bogue, le chroot ne sera pas compromis. Remarquez que, contrairement au cas de la configuration d'un chroot par utilisateur, le démon SSH est exécuté dans le même chroot que les utilisateurs, donc il y a au moins un processus potentiel exécuté en tant que superutilisateur qui pourrait s'échapper du chroot.

Remarquez aussi que pour permettre à SSH de fonctionner à cet endroit, la partition où le répertoire du chroot existe ne doit pas être montée avec l'option *nodev*. Avec cette option, l'erreur suivante se produira : *PRNG is not seeded* car `/dev/urandom` ne fonctionne pas dans le chroot.

Configuration d'un système minimal (la manière vraiment simple)

Utilisez debootstrap pour configurer un environnement minimal qui n'inclut que le serveur SSH. Pour faire cela, il suffit de créer un chroot comme décrit dans la http://www.debian.org/doc/manuals/reference/ch09#_chroot_system. Cette méthode est sûre de fonctionner (tous les composants nécessaires seront dans le chroot) mais coûtera de l'espace disque (une installation minimale de Debian représente plusieurs milliers de mégaoctets). Ce système minimal pourrait aussi intégrer des fichiers setuid qu'un utilisateur dans le chroot pourrait utiliser pour s'en échapper si l'un d'entre eux peut être utiliser pour une augmentation de droits.

Créer l'environnement automatiquement (la manière simple)

Vous pouvez facilement créer un environnement restreint avec le paquet makejail puisqu'il prend automatiquement soin de tracer le démon serveur (avec **strace**), et l'exécute dans l'environnement restreint.

L'avantage de programmes qui génèrent automatiquement l'environnement de **chroot** est qu'ils sont capables de copier tout paquet vers l'environnement de **chroot** (en suivant même les dépendances de paquet et en s'assurant qu'il est complet). Fournir les applications aux utilisateurs est donc plus facile.

Pour mettre en place l'environnement en utilisant les exemples fournis par **makejail**, créez simplement `/var/chroot/sshd` et exécutez la commande suivante :

```
# makejail /usr/share/doc/makejail/examples/sshd.py
```

Cela configurera la chroot dans le répertoire `/var/chroot/sshd`. Remarquez que ce chroot ne sera pas complètement fonctionnel avant de :

- monter le système de fichiers *procfs* dans `/var/chroot/sshd/proc`. **makejail** le montera tout seul, mais si le système redémarre, il faudra le remonter en exécutant :

  ```
  # mount -t proc proc /var/chroot/sshd/proc
  ```

  Il peut aussi être monté automatiquement en modifiant `/etc/fstab` pour ajouter cette ligne :

  ```
  proc-ssh /var/chroot/sshd/proc  proc none 0 0
  ```

- faire écouter syslog sur le périphérique `/dev/log` dans le chroot. Pour faire cela, il faut modifier `/etc/default/syslogd` pour ajouter *-a /var/chroot/sshd/dev/log* à la définition de la variable *SYSLOGD*.

Consultez le fichier d'exemple pour savoir quels autres modifications doivent être réalisées dans l'environnement. Certaines de ces modifications, comme la copie des répertoires personnels des utilisateurs, ne peuvent être réalisés automatiquement. Limitez également l'exposition des informations sensibles en ne copiant que les données d'un nombre donné d'utilisateurs des fichiers `/etc/shadow` ou `/etc/group`. Remarquez qu'en utilisant la *séparation de droits*, l'utilisateur *sshd* doit exister dans ces fichiers.

L'environnement d'exemple suivant a été (légèrement) testé dans Debian 3.0 et est construit avec le fichier de configuration fourni par le paquet et inclut le paquet 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 répertoire, 733 fichiers
```

Avec Debian 3.1, il faut s'assurer que l'environnement inclus aussi les fichiers communs pour PAM. Les fichiers suivants doivent être copiés dans le chroot si **makejail** ne l'a pas fait:

```
$ 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
```

Créer soi-même l'environnement (la manière difficile)

Il est possible de créer un environnement, en utilisant une méthode d'essai-et-d'erreur, en surveillant les traces du serveur **sshd** et les fichiers journaux pour déterminer les fichiers nécessaires. L'environnement suivant, fourni par José Luis Ledesma, est un listing exemple des fichiers dans un environnement de **chroot** pour **ssh** dans Debian Woody (3.0) : [9]

```
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/
```

[9] Remarquez qu'il n'y a pas de fichiers SETUID. Cela rend plus difficile pour les utilisateurs distants de s'échapper de l'environnement de **chroot**. Cependant, il empêche également les utilisateurs de changer leurs mots de passe, car le programme **passwd** ne peut pas modifier les fichiers `/etc/passwd` ou `/etc/shadow`.

```
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//
```

Environnement de chroot pour Apache

Introduction

L'utilitaire **chroot** est souvent utilisé pour emprisonner un démon dans une arborescence restreint. Vous pouvez l'utiliser pour isoler des services d'autres services, pour que les problèmes de sécurité d'un paquet logiciel ne mettent pas en péril le serveur tout entier. Quand vous utiliser le script **makejail**, mettre en place et mettre à jour l'arborescence chrooté est beaucoup plus facile.

FIXME : Apache peut aussi être chrooté en utilisant http://www.modsecurity.org qui est disponible dans libapache-mod-security (pour Apache 1.x) et libapache2-mod-security (pour Apache 2.x).

Licence

Installer le serveur

Cette procédure a été testée sur Debian GNU/Linux 3.0 (Woody) avec **makejail** 0.0.4-1 (de Debian/testing).

- Connectez-vous en tant que **root** et créez le nouveau répertoire prison :

```
$ mkdir -p /var/chroot/apache
```

- Créez un nouvel utilisateur et un nouveau groupe. Le serveur Apache chrooté fonctionnera sous cet utilisateur et groupe, qui n'est utilisé pour rien d'autre sur le système. Dans cet exemple, l'utilisateur et le groupe sont appelés **chrapach**.

```
$ adduser --home /var/chroot/apache --shell /bin/false \
--no-create-home --system --group chrapach
```

FIXME : Est-ce qu'un nouvel utilisateur est nécessaire ? (Apache fonctionne déjà sous l'utilisateur apache)

- Installez Apache comme d'habitude sous Debian : `apt-get install apache`
- Configurez Apache (par exemple définissez les sous-domaines, etc.). Dans le fichier de configuration `/etc/apache/httpd.conf`, positionnez les options *Group* et *User* à `chrapach`. Redémarrez Apache et assurez-vous que le serveur fonctionne correctement. Maintenant, stoppez le démon Apache.
- Installez **makejail** (disponible dans Debian/testing actuellement). Vous devriez également installer **wget** et **lynx** car ils sont utilisés par **makejail** pour tester le serveur chrooté : `apt-get install makejail wget lynx`
- Copiez le fichier de configuration exemple pour Apache dans le répertoire `/etc/makejail` :

```
# cp /usr/share/doc/makejail/examples/apache.py /etc/makejail/
```

- Éditez `/etc/makejail/apache.py`. Vous devez changer les options *chroot*, *users* et *groups*. Pour exécuter cette version de **makejail**, vous pouvez également ajouter une option **packages**. Consultez la http://www.floc.net/makejail/current/doc/. Un exemple est fourni ici :

```
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 : Certaines options semblent ne pas fonctionner correctement. Par exemple, `/etc/shadow` et `/etc/gshadow` ne sont pas copiés, alors que `/etc/password` et `/etc/group` sont intégralement copiés au lieu d'être filtrés.

- Créez l'arborescence de chroot : `makejail /etc/makejail/apache.py`
- Si les fichiers `/etc/password` et `/etc/group` ont été intégralement copiés, entrez :

```
$ grep chrapach /etc/passwd > /var/chroot/apache/etc/passwd
$ grep chrapach /etc/group > /var/chroot/apache/etc/group
```

  pour les remplacer avec des copies filtrées.

- Copiez les pages du site web et les journaux dans la prison. Ces fichiers ne sont pas copiés automatiquement (consultez l'option *preserve* du fichier de configuration de **makejail**).

```
# cp -Rp /var/www /var/chroot/apache/var
```

```
# cp -Rp /var/log/apache/*.log /var/chroot/apache/var/log/apache
```

- Éditez le script de démarrage pour le démon de journaux système pour qu'il écoute également sur la socket `/var/chroot/apache/dev/log`. Dans `/etc/default/syslogd`, remplacez : `SYSLOGD=""` par `SYSLOGD=" -a /var/chroot/apache/dev/log"` et redémarrez le démon (`/etc/init.d/sysklogd restart`).

- Éditez le script de démarrage d'Apache (`/etc/init.d/apache`). Vous pouvez avoir besoin d'effectuer certaines changements au script de démarrage par défaut pour qu'il fonctionne correctement dans une arborescence chrooté. Comme :

  - configurer une nouvelle variable *CHRDIR* au début du fichier ;

  - éditer les sections *start*, *stop*, *reload*, etc. ;

  - ajouter une ligne pour monter et démonter le système de fichiers `/proc` dans la prison.

```
#! /bin/bash
#
# apache        Start 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 \
      --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" \
      --signal USR1 --startas $DAEMON --chroot $CHRDIR
    ;;

  reload-modules)
    echo -n "Reloading $NAME modules"
    start-stop-daemon --stop --pidfile "$CHRDIR/$PIDFILE" --oknodo \
      --retry 30
    start-stop-daemon --start --pidfile $PIDFILE \
      --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
  echo .
  exit 0
else
  echo failed
  exit 1
fi
```

FIXME : Est-ce que le premier processus Apache devrait être lancé avec un autre utilisateur que root (c'est-à-dire ajouter --chuid chrapach:chrapach) ? Désavantage : chrapach devra avoir un accès en écriture aux journaux, ce qui est étrange.

- Remplacez dans `/etc/logrotate.d/apache` `/var/log/apache/*.log` par `/var/chroot/apache/var/log/apache/*.log`

- Démarrez Apache (**/etc/init.d/apache start**) et vérifiez ce qui est indiqué dans les journaux de la prison (`/var/chroot/apache/var/log/apache/error.log`). Si votre configuration est plus

complexe (e.g. si vous utilisez également PHP et MySQL), des fichiers seront probablement manquants. Si certains fichiers ne sont pas copiés automatiquement par **makejail**, vous pouvez les indiquer dans les options *forceCopy* (pour copier les fichiers directement) ou *packages* (pour copier les paquets en entier et leurs dépendances) du fichier de configuration `/etc/makejail/apache.py`.

- Entrez `ps aux | grep apache` pour vous assurer qu'Apache fonctionne. Vous devriez voir quelque chose comme :

```
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
```

- Assurez-vous que les processus Apache fonctionnent bien dans le chroot en observant le système de fichiers `/proc` : `ls -la /proc/numero_processus/root/.` où `numero_processus` est l'un des numéros de PID de la liste ci-dessus (2e colonne ; 189 par exemple). La liste des entrées pour une arborescence restreinte devraient être ainsi :

```
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
```

  Pour automatiser ce test, vous pouvez entrer :`ls -la /proc/`cat /var/chroot/apache/var/run/apache.pid`/root/.`

  FIXME : Ajouter d'autres tests qui peuvent être exécuter pour s'assurer que la prison est fermées ?

La raison pour laquelle j'aime cela est que la mise en place d'une prison n'est pas très difficile et que le serveur peut être mis à jour avec seulement deux lignes :

```
apt-get update && apt-get install apache
makejail /etc/makejail/apache.py
```

Consultez également

Si vous recherchez plus d'informations, vous pouvez envisager ces sources d'informations sur lesquelles les informations présentées sont basées :http://www.floc.net/makejail/, ce programme a été écrit par Alain Tesio